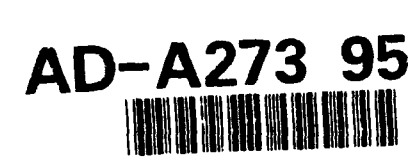

ARMY RESEARCH LABORATORY

Nuclear-Survivability Cost Analysis Using Fuzzy-Set Theory

Aivars Celmiņš

ARL-TR-300 November 1993

NOTICES

Destroy this report when it is no longer needed. DO NOT return it to the originator.

Additional copies of this report may be obtained from the National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Road, Springfield, VA 22161.

The findings of this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

The use of trade names or manufacturers' names in this report does not constitute indorsement of any commercial product.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	November 1993	Final, Jun 92–May 93

4. TITLE AND SUBTITLE
Nuclear-Survivability Cost Analysis Using Fuzzy-Set Theory

5. FUNDING NUMBERS
PR: 1L162618AH80

6. AUTHOR(S)
Aivars Celmiņš

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
ATTN: AMSRL-CI-CA
Aberdeen Proving Ground, MD 21005-5067

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
ATTN: AMSRL-OP-CI-B (Tech Lib)
Aberdeen Proving Ground, MD 21005-5066

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
ARL-TR-300

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

To conduct a cost-effectiveness analysis of the nuclear survivability of weapon systems, one needs to evaluate and compare different proposed system modifications. That task can require the evaluation of a large number of options and the processing of vague information. This report describes a pilot computer program that has been developed to assist the decision maker in such tasks. The program treats vague information using the formalism of fuzzy set theory and provides the decision maker with a set of best options according to criteria specified by the user. The purpose of the pilot program was to develop and test numerical methods for the treatment of vague data for this application and to establish the feasibility of the general approach for the analysis of survivabilty. It was found that fuzzy set theory can be used to assist cost-effectiveness analysis of nuclear survivability and that the approach taken in the pilot program can be used with minor modifications in the development of full scale utility programs.

14. SUBJECT TERMS	15. NUMBER OF PAGES
cost operational effectivenes analysis; fuzzy sets; nuclear survivability; nuclear hardening; cost analysis	103
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

Page

LIST OF FIGURES v
LIST OF TABLES vii
ACKNOWLEDGMENTS ix
1. INTRODUCTION 1
2. PROGRAM OUTLINE 2
3. DATA BANK 4
3.1. Purpose of the Data Bank 4
3.2. Description of the System 4
3.3. Element Response to the Environment 5
4. CASE INPUT 5
4.1. Contents of the Input File 5
4.2. Environment 6
4.3. Actions 7
4.4. Inquiries 8
5. CURRENT DATA BASE WITH EXIT SURVIVABILITIES 9
6. ANALYSIS OF ACTIONS AND OPTIONS 10
7. SELECTION OF BEST OPTIONS 11
7.1. Interpretation of System Survivabilities 11
7.2. Ranking of System Survivabilities 12
8. EXAMPLES 13
8.1. Three-Element Systems 13
8.2. One-Element Many-Option System 16
9. SUMMARY AND CONCLUSIONS 18
10. REFERENCES 21
Appendix A.
COMPUTATION OF BASE–LINE SURVIVABILITIES OF ELEMENTS 23
Appendix B. LOGICAL COMBINATIONS OF SURVIVABILITIES 29
Appendix C. LIST OF THE PROGRAM SCAP 35
Appendix D. DATA BANK ENTRY FOR THE "SYSTEM 3a" 87
Appendix E. CASE INPUT FOR THE "SYSTEM 3a" 91
Appendix F. SUMMARY OF INPUT FOR THE "SYSTEM 3a" 95
DISTRIBUTION LIST 99

INTENTIONALLY LEFT BLANK

LIST OF FIGURES

Figure		Page
1.	Element-survivability curve	6
2.	Memberships of hardening categories	8
3.	Memberships of system-survivability categories	12
4.	Equivalent Systems 3a and 3b	13
5.	Survivability of System 3a with option No. 4	15
6.	Survivability of System 3a with option No. 1	15
7.	Equivalent Systems 3C and 3D	16
8.	Survivability of System 3C with option No. 5	17
9.	Survivability of System 3C with option No. 6	17
10.	Survivability of System 4001 with option No. 127	19
A1.	Fuzzy element-survivability curve with level lines	25
A2.	Interpolation intervals	26
A3.	Derivation of interpolation formulas	27
B1.	Conjunctive system	31
B2.	Conjunctive combination of survivabilities	32
B3.	Disjunctive system	33
B4.	Disjunctive combination of survivabilities	33

INTENTIONALLY LEFT BLANK

LIST OF TABLES

Table		Page
1.	Outline of the Main Program	3
2.	Environment Parameters	6
3.	Categories of Cost Changes	7
4.	Categories of Survivability Changes	8
5.	Categories of System Survivability	12
6.	Best Options for Overall Survivability of System 3a	14
7.	Best Options for Partial Survivability of System 3C	16
8.	Problem Formulation for a Simple System Hardening	18

INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

The author thanks Mr. Albert Gluckman, Survivability/Lethality Analysis Directorate of the U.S. Army Research Laboratory for valuable suggestions and discussions of the nuclear survivability cost analysis problem. He also thanks Mr. Keith Warner from the same Directorate for continued interest and support in the development of the program.

INTENTIONALLY LEFT BLANK

1. INTRODUCTION.

Cost-effectiveness analyses in the area of nuclear warfare require among other tasks the estimation of nuclear survivabilities of weapon systems. In particular, to make rational recommendations about system improvements, one needs estimates of system-survivability changes and associated cost changes that would result from hardening or softening of some elements of the system. If the system is simple and the number of proposed changes is small, then such estimates can be easily obtained. If, however, the system consists of many elements and a large number of modifications is proposed, then the assistance of a computer becomes necessary because the number of combinations of proposed modifications grows exponentially with the number of proposals. Some of the information about the system, its hardening, and the costs of hardening typically is only approximate. Therefore, a supporting computer program must be able to handle large amounts of information that consist of a mixture of exact and approximate data.

To explore the applicability of computer assistance to cost-effectiveness analyses, the author developed a computer program that calculates changes in the survivability of a weapon system due to modifications of individual elements of the system and provides a simple analysis of the results. This report describes the pilot program. The purpose of the pilot program is, first, to test the usefulness of such a program as a management tool and, second, to ascertain by practical experience the best methods for handling vague information by a computer. The program computes system survivabilities for all proposed modifications, prepares a list of the results, inspects the list, and prepares another list containing the best modifications according to criteria specified by the user. We allow some of the input information to be vague or linguistic. For instance, costs may be specified as "high" and survivability increase as "medium" for some elements. Working with this type of information can be handy in the early stages of developing hardening strategies. Linguistic information of the described type can be treated in a rational manner with the aid of fuzzy-set theory (see Zadeh 1965, Zimmermann 1991, or Klir and Folger 1988). According to that theory, linguistic information is represented by fuzzy sets. Consequently, the data that are analyzed by the program can be either crisp (exact numbers) or fuzzy (approximate numbers). The output of the program consists of the aforementioned list of best options with corresponding estimates of system survivability and costs. These estimates generally are fuzzy sets (if some of the input is linguistic or fuzzy) and are translated by the program into linguistic terms to assist in the interpretation of the results.

The pilot program for the described task is called **Scap** and is coded in Fortran 77. A short outline of the program is given in Section 2. The input to the program

consists of two parts and is described in Sections 3 and 4. The two input parts are, first, a description of the structure and present state of the weapon system that is retrieved from a data bank and, second, the case input that contains the proposed hardening/softening actions and is stored in an input file. In Sections 5 through 7 we outline the logic of the program and the methods for treating the mixture of crisp and fuzzy data. Examples of calculations with Scap are given in Section 8. Section 9 contains a summary and conclusions.

2. PROGRAM OUTLINE.

We consider weapon systems that are combinations of a set of interdependent system elements. The computer program Scap computes the present survivability of the weapon system from the present survivabilities of its elements, and estimates of the survivability of the modified system from proposed modifications of the element survivabilities. The computations are based on the following data.

(I.1) A description of the system in terms of its elements showing the interdependency among the elements in a logical fault tree. Such descriptions are generally available for major weapon systems, in particular if a survivability analysis has been conducted for the system. In general, the descriptions are established by audition of experts. We assume that this information can be retrieved from a data bank.

(I.2) Present survivabilities of each element of the system with respect to all threat environments of interest. Again, such data are available for systems whose survivabilities have been analyzed. This information, too, is assumed to be available from a data bank.

(I.3) Values of those environment parameters that are of interest for the investigation. This and the following are "case inputs" for a specific run of the program.

(I.4) Proposed "actions", that is, survivability modifications (hardenings or softenings) of subsets of system elements with corresponding cost changes (cost increases or savings).

Scap is programmed to answer questions such as "How does the survivability of the system change in the given environment if moderate costs are invested to slightly harden the system elements A and B ?" Answers to such questions enable the decision maker either to make a reasonable decision directly or to feed the answers into an expert system and obtain from it suggestions for a decision. If several system modifications are proposed, then the decision maker must be able to choose rationally between various combinations of the proposed modifications. Because the number of combinations of modifications increases exponentially with the number of modifications, one needs, even for relatively simple sets of modifications, the assistance of a computer for finding the best among all possible combinations. Scap can assist the decision maker

Table 1. Outline of the Main Program

Read current problem description	Current values of environment parameters (16 values). System identification. Actions (hardness changes of elements & cost changes). Inquiries.
Consult general data bank	Read from a general data file the specifications of all elements of the system.
Make a current data base	Compute element and system survivabilities for the current environment.
Analyze options	Loop over all combinations of proposed actions: (1) Compute element survivabilities for the current combination of actions. (2) Compute corresponding system survivability. (3) Store: Action combinations and their costs; ID numbers of affected elements; System survivabilities (16 values).
Summarize analyses	Combine results for individual parameters as specified by the inquiries. Store results in output files.
Find best options	Find the cheapest options that produce high lower bounds of system survivabilities. Store results in output files.

by providing a list of those options that are worthy of further consideration.

The queries that are accommodated by Scap are

(Q.1) Survivability of the system for all possible combinations of proposed actions and with respect to each environment parameter. The answer to this query is a complete list of all possible outcomes, but the list is generally useless because of its length.

(Q.2) Five highest survivabilities of the system, computed with respect to a specified

subset of environment parameters.

(Q.3) Five highest overall survivabilities of the system, i.e., the highest combined survivabilities with respect to all environment parameters.

The input to Scap can be in the form of crisp or fuzzy numerical data or in the form of linguistic information. The output is fuzzy except in cases when all input is crisp. The printed output contains also a linguistic interpretation of the numerical results. Scap generates three forms of output: a printable list that can be studied by the decision maker, a computer file containing a comprehensive list of results that can be used as input for other analysis programs, and graphical output consisting of plots of membership functions of selected system survivabilities.

A schematic overview of the main program of Scap is shown in Table 1. Appendix C contains a complete listing of the program.

3. DATA BANK.

3.1. Purpose of the Data Bank.

We assume that baseline properties of weapon systems can be obtained from a nuclear-survivability data bank. To test the pilot program, such a data bank was generated in the form of a computer file containing descriptions of several hypothetical systems. The systems were described by their functional fault trees in terms of their elements, and elements were described in terms of their responses (survivabilities) to environment parameters. These data permit one to compute the baseline properties of the system, that is, estimates of the present survivability for specified ranges of environment parameter values. This section describes the data that are stored in the data bank. As an example for the structure of the data bank, we present in Appendix D the actual contents of that part of the data-bank file that describes a hypothetical "System 3a" used in the sample calculations of Section 8.1.

3.2. Description of the System.

The system is defined in terms of its elements by a fault-tree structure. The description of the fault tree is stored in a data bank that contains for each system a list of its elements, and for each element an information frame. (In the pilot program Scap the number of elements in a system must be less or equal 12.) If n is the number of elements, then the system itself is listed as an element with the number $n + 1$. Each element-information frame contains the following data:

(E.1) An identification number and an alphanumeric name of the element.

(E.2) A list of consequences, i.e., a list of those elements whose functioning directly depends on this element. (For the element representing the system, this list is of course empty.)

(E.3) A list of conjunctive antecedents, i.e., a list of all those elements that must

function simultaneously and independently for this element to function. In a logical-gate representation, these elements would be connected to the given element by an "and" gate. The numerical computation of the conjunction of element survivabilities is described in Appendix B.

(E.4) A list of disjunctive antecedents, i.e., a list of all those elements that are sufficient for the functioning of the given element. In a logical-gate representation, these elements would be connected to the given element by an "or" gate. The disjunctive combination of element survivabilities is described in Appendix B. If the element has also conjunctive antecedents listed in (E.3) then their combined output is disjunctively combined with the combined output of the disjunctive antecedents to form the input for the element.

(E.5) A description of the response of the element to the environment. This is a collection of 16 functions describing the dependency of the element's survivability on each of 16 environment parameters that define the threat environment (see Table 2). The response functions can be either crisp or fuzzy. The functions in the present data bank are restricted to a class of linear functions as described in Section 3.3.

3.3. Element Response to the Environment.

The current environment is defined by values or value ranges of the 16 environment parameters listed in Table 2. To simplify the calculations we assume that all environment parameters are normalized to the interval [1,100]. The survivability of each element with respect to an environment parameter is computed by interpolation in the corresponding element-survivability function that is obtained from the data bank. In the present data bank, each function is defined by six function parameters t_1, t_2, t_3, b_1, b_2, and b_3, as shown in Figure 1. The solid center curve in Figure 1 represents the core of the fuzzy survivability function, that is, the points where the membership value of the function equals unity. The dashed outer curves show the outline of the support, that is, the outline of the area where the survivability function has a positive membership value. If the three curves coincide, then they define a crisp survivability function; otherwise, they define a fuzzy function. To calculate for a given value of the environment parameter the survivability value of the element, we interpolate in this function using the algorithm described in Appendix A. The response of the element to a specific environment consists of 16 survivability values (crisp of fuzzy), one for each environment parameter, obtained by the interpolation algorithm.

4. CASE INPUT.

4.1. Contents of the Input File.

The case-input file for Scap contains the specification of the environment for which survivability estimates should be computed, a list of the actions that are to be

Table 2. Environment Parameters.

1. Over-pressure peak	9. Total dose, silicon
2. Over-pressure impulse	10. Total neutron dose
3. Dynamic-pressure peak	11. Neutron fluence
4. Dynamic-pressure impulse	12. Total gamma dose
5. Under-pressure peak	13. Minimum threat yield
6. Total thermal energy	14. Maximum threat yield
7. Maximum irradiance	15. Ex-atmospheric EMP
8. Total dose, tissue	16. Endo-atmospheric EMP

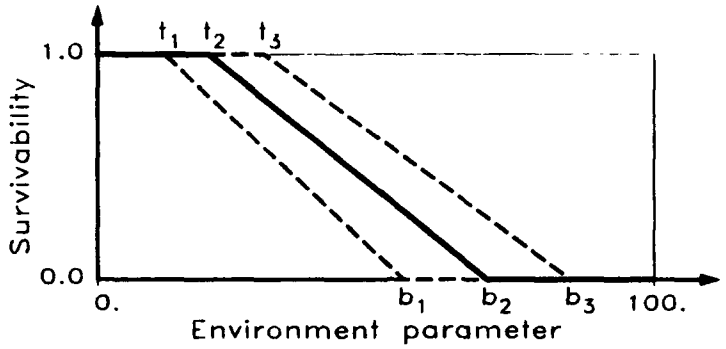

Figure 1. Element-survivability curve.

investigated, a list of queries, and a list of those system-survivability membership functions that are to be plotted. This section provides a comprehensive description of the input. A listing of an actual case-input file for a hypothetical "System 3a" (see Section 8.1) is shown in Appendix E. Appendix F is generated by Scap and contains a summary of all input (data-bank and case input) for the same example.

4.2. Environment.

The environment for each case study is defined by the values of the environment parameters. In the pilot program Scap, these values are assumed to be either crisp numbers or fuzzy numbers with triangular membership functions. The triangular membership functions are specified in the input by three crisp membership parameters: the left-hand abscissa p_1 of the support of the fuzzy number (i.e., the abscissa of the left end of the triangle base), the abscissa p_2 of the core of the fuzzy number (i.e., the abscissa of the apex of the triangle), and the right-hand abscissa p_3 of the support. If the three membership parameters are equal, then the set defines a crisp number. The apex abscissa p_2 must be within the interval [0,100], because all environment parameters were assumed to be normalized to that interval. In summary, the environment is characterized by 16 three-number sets. In a final utility program, one will of course allow more general environment specifications, e.g., by environment

parameters that are trapezoidal fuzzy numbers or, more generally, are fuzzy numbers defined by a list of membership values.

4.3. Actions.

The purpose of the program Scap is to analyze proposals of system modifications that consist of nuclear hardening or softening of some elements of the system. We call these proposals *actions*. Actions may be proposed by manufacturers of the system, suggested by some engineering breakthrough, or be part of a "what if" study. To make a cost analysis of the proposals possible, the description of each action must contain cost estimates of the modifications. This information is important because the changes of element properties and the corresponding cost changes do not necessarily have the same trend, for instance, when an element can be replaced by a harder and cheaper element. In such cases the decision maker would like to know whether the effect of the replacement on the survivability of the whole system is sufficiently large to justify associated savings or expenses. (Even when the new elements are cheap, their replacement in existing systems can be time consuming and costly.)

The Scap program can analyze up to seven independent actions. Each action is described in the program input by specifying changes of system costs, providing a list of affected elements, and giving for each affected element the change of its survivability with respect to each of the 16 environment parameters. The cost changes are assumed to be linguistic input, restricted to one of the seven categories listed in Table 3.

Table 3. Categories of Cost Changes.

large savings	small increase
medium savings	medium increase
small savings	large increase
none	

A modification of the nuclear survivability is defined by providing for each affected element 16 survivability changes corresponding to the 16 environment parameters. These changes are assumed to be describable by one of the 11 categories listed in Table 4. (The choice of 11 categories is arbitrary but it is consistent with the six survivability categories listed in Table 5, page 12, in the sense that a "small" change transforms any survivability category into the next higher or lower category.) Table 4 lists the names of the survivability-change categories and the corresponding numerical values of the survivability changes. These values are fuzzy, except for the categories "no change", "negative large", and "positive large". In the case of a negative large change (large softening), the change of the survivability is a crisp "-1", and it makes the element not-surviving for any value of the particular environment parameter (survivability zero). The positive large change (large hardening) equals the crisp change "$+1$" and makes the element invulnerable, that is, the element is assigned a survivability value of unity for any value of the particular environment parameter. The

membership functions of the fuzzy hardening categories are displayed in Figure 2. The negative changes, i.e., the softening categories, have corresponding membership functions on the negative axis of survivability changes. In a final version of the program, one would of course allow also crisp changes of survivability, but in the pilot program the input is restricted to fuzzy changes because the author was mainly interested in testing the applicability of fuzzy logic and fuzzy arithmetic to the present problem.

Table 4. Categories of Survivability Changes.

Category	Value	Category	Value
Negative large	-1	Positive small	about 0.2
Negative large medium	about -0.8	Positive small medium	about 0.4
Negative medium	about -0.6	Positive medium	about 0.6
Negative small medium	about -0.4	Positive large medium	about 0.8
Negative small	about -0.2	Positive large	1
No change	0		

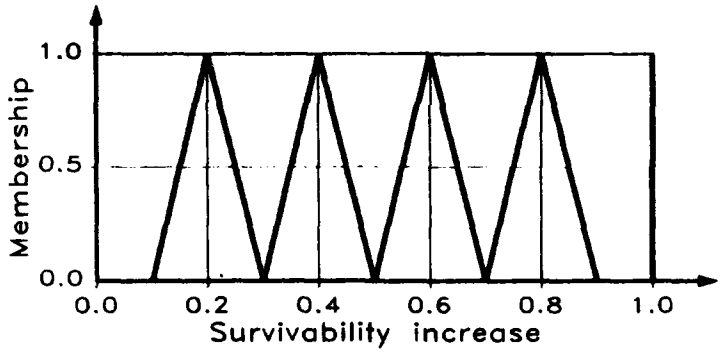

Figure 2. Memberships of hardening categories.

4.4. Inquiries.

The three inquiries listed in Section 2 appear in the Scap input file as a three-component query vector. Each component of the vector corresponds to one of the three queries (Q.1), (Q.2) and (Q.3), respectively. If the first component is non-zero then an output file will be generated containing the survivability of the system for all combinations of actions and with respect to each parameter. Let the number of proposed independent actions be N. Then the total number of different survivabilities due to combinations of actions is $16 \cdot (2^N - 1)$. Therefore, this output is meaningful only if the number of proposed actions is small.

The second component of the inquiry vector instructs Scap to produce a list of five options with the highest system survivabilities with respect to a subset of parameters.

The value of the component must be between 0 and 16, and it indicates the number of parameters in the subset. If the number is zero, then no list is produced. If the number is positive, then the subset itself also must be included in the input. The combined survivability with respect to the subset of parameters is computed by a conjunctive combination of the survivabilities with respect to each parameter in the subset.

A non-zero third component of the inquiry vector instructs Scap to make a list of five options that have the highest system survivabilities with respect to all parameters. This overall system survivability is computed by a conjunctive combination of the survivabilities with respect to each of the 16 environment parameters.

All queries can be activated simultaneously, that is, the answers to all queries are produced by the same computer run. The output consists of two files for each query. The first file is in printable format and it contains the answers to the inquiry in a form that is easy to read. The second file contains the same information in simpler form and is meant as input for future programs that analyze the output of Scap.

Requests for graphical output are placed in the final portion of the input file. The requests generate plots of membership functions of the system survivability and consist of a list of three-number sets. The three numbers specify, respectively, the inquiry type, the option number and the number of the environment parameter with respect to which the survivability should be plotted. (The parameter number is relevant for inquiries of Type (Q.1) when system survivability with respect to individual parameters are of interest.) The numbers of those options for which plotting is requested can be obtained from the printed output lists. That is, we assume that usually the plots will be requested in a second run of the program if additional information about an interesting option is deemed necessary. The request for plots causes Scap to produce a file with the data that are necessary for plotting. The actual plotting is done by a separate plotting program that reads the file generated by Scap and uses the graphical system DI-3000 of Precision Visuals, Inc., to generate the plots.

5. CURRENT DATA BASE WITH EXIT SURVIVABILITIES.

The case input contains a system identification label. Using that label, Scap retrieves the description of the system from a data bank. The retrieved description enables Scap to compute the survivabilities of all elements of the system for the present environment specified by the case input, that is, 16 survivabilities for each element. The computed survivabilities can be either crisp or fuzzy numbers and are stored as fuzzy sets. The algorithm for the calculation of the survivabilities is described in Appendix A. The collection of all element survivabilities constitutes the *current data base.*

The current system survivability is computed as follows from the data in the current data base. First, the element that represents the system is assigned survivabilities equaling unity, and all elements are labeled as not-completed. The

program then assigns to each element without antecedents exit survivabilities that equal the current data base survivabilities of the element. Elements with exit survivabilities are labeled as completed. Next, the program scans through all completed elements with not-completed consequences and checks whether the combined entry survivabilities of any of the consequential elements can be computed. (The computation of the combined entry survivabilities is not possible if the consequential element has some antecedents that are not completed. In that case nothing is computed and the consequential element remains labeled as not-completed.) If the computation is possible, then the exit survivabilities of the consequential element are computed as follows for each environment parameter. First, the exit survivabilities of the antecedents are combined to establish an entry survivability of the consequential element. The combination of the antecedent survivabilities is either conjunctive or disjunctive according to the specifications in the element-information frame. The algorithms for the logical combination of survivabilities are described in Appendix B. Next, the combined entry survivability is conjunctively combined with the survivability of the consequential element itself to produce the exit survivability of the element. When all 16 exit survivabilities of an element are computed, then the element is labeled as completed. The scan is repeated through all completed elements with not-completed consequences until the exit survivabilities of the element representing the system itself are computed. Those survivabilities are taken as the system survivabilities of the current data base. This process requires at most as many scans as there are elements in the system. This means for the pilot program that at most 12 scans are needed, because in that program, the maximum number of system elements is 12.

6. ANALYSIS OF ACTIONS AND OPTIONS.

The input contains a list of actions with proposed system modifications. Each action is defined by a set of system elements and for each element in the set, a proposed change of the survivabilities with respect to each of the 16 environment parameters. We assume that all actions are independent and can be combined. A combination of actions we call an *option*. Scap is programmed to handle up to seven actions which means that the program can analyze up to 127 options (combinations of actions). After reading the input, Scap establishes a list of all options and computes the system survivability for each option and with respect to each of the 16 environment parameters. (Hence, up to $16 \cdot 127 = 2032$ membership functions of the system survivability might be stored during a computer run.) The algorithm for the computation of the system survivability for any given option is the same (and is done in the same subroutine) as that for the system survivability of the current data base. The algorithm is described in Section 5. The outcomes are different for different options because the survivabilities of the elements are changed by the actions that constitute an option. The new element survivabilities are computed by adding the element-survivability changes from the input to the element survivabilities of the current data

base. This involves the addition of two fuzzy numbers under the constraint that the result must be within the interval [0,1]. Algorithms for fuzzy-number arithmetic are described, e.g., by Kaufmann and Gupta 1985.

If an option includes the hardening or softening of the same element by several actions, then Scap adds the survivability changes consecutively. This rule was chosen for simplicity and it might not be appropriate for all problems. In general, a combination of independent actions that affect the same elements is suspect and requires special investigation of any particular case. A final utility program should attach a warning label to options that contain such combinations.

7. SELECTION OF BEST OPTIONS.

7.1. Interpretation of System Survivabilities.

Scap computes for each option of system modifications the fuzzy values (i.e., the membership functions) of the relevant system survivabilities. (The relevant survivabilities are defined by the inquiries described in Section 4.4.) To aid in the interpretation of a computed fuzzy survivability, Scap provides a linguistic description of it in terms of the six survivability categories that are listed in Table 5. (Six categories were chosen arbitrarily. A finer or coarser granulation can be easily implemented.) The table contains the names of the categories and the fuzzy survivability values associated with each category. Figure 3 shows the membership functions of the survivability values corresponding to the six survivability categories. The membership function of a system survivability that is computed by Scap generally will not exactly match any of these membership functions. Hence, the task is to select one or more survivability categories of Table 5 that approximately describe the survivability membership function. If the approximation with one category is not sufficient, for instance, when the membership function of the system survivability has a broad support, then Scap specifies lower and upper bounds of the survivability in terms of the same six standard categories. We now describe the algorithm for the determination of the proper categories.

Let A be the area of intersection between two membership-function triangles in Figure 3 with a separation of 0.1 between their apexes. (The separation 0.1 equals one half the distance between the apexes of any two adjacent triangles in Figure 3.) To interpret a system survivability that is given by its membership function, Scap first computes the areas of intersection between the area under the system-survivability membership curve and each of the six membership-function triangles. The leftmost category with an intersection area larger or equal to A is denoted as the lower bound of the survivability of the system. The rightmost category with the same property is denoted as the upper bound of the survivability of the system. Using these matches the system survivability is described in linguistic terms, for instance, as "moderate" or "poor to quite good", etc. If none of the intersections is larger than A then the system

Table 5. Categories of System Survivability.

Survivability Category	Survivability Value
Very poor	about 0.0
Poor	about 0.2
Moderate	about 0.4
Quite good	about 0.6
Good	about 0.8
Very good	about 1.0

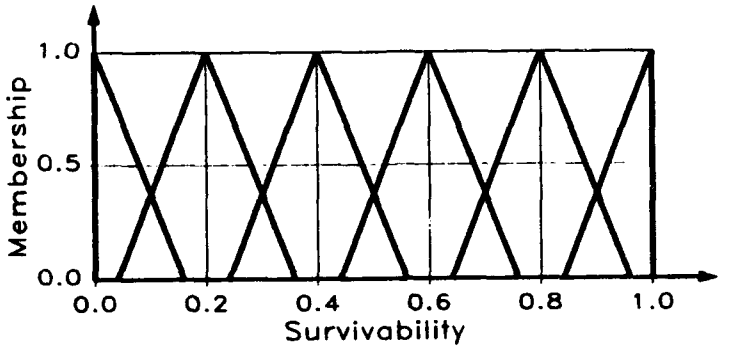

Figure 3. Memberships of system-survivability categories.

survivability is assigned the category with the largest intersection.

If the system survivability is crisp, then Scap assigns to it the linguistic label of that category which is nearest to the value of the survivability. For crisp system survivabilities with the values zero and unity, Scap uses the special linguistic labels "destroyed" and "unaffected", respectively.

7.2. Ranking of System Survivabilities.

The principal result provided by Scap is a set of system-survivability membership functions, each corresponding to an option of system changes. To analyze this result, we want to order the survivabilities according to their size. The ranking of fuzzy numbers is, however, not unique and many ranking methods have been proposed. Examples of recent articles about the ranking of fuzzy sets are Bortolan and Degani 1985, Kim and Park 1990, Dubois and Prade 1991, and Choobineh and Li 1993. Every ranking method typically is designed for a special type of membership function, and for a particular application. In our application, the fuzzy numbers (survivabilities) that we want to rank are characterized by the following properties.

(1) The membership functions are general, i.e., not restricted to a special type,

such as triangular, trapezoidal, etc.

(2) The most important part of each membership function is its lower end, that is, the minimum of possible survivability.

(3) The supports of the survivability values are restricted to the interval [0,1].

The author is not aware of any published ranking method that is applicable to general membership functions and emphasizes the lower end of the membership function. Therefore, for the present problem, a ranking method was developed based on an interpretation of the survivability in terms of survivability categories. In this ranking method, we use as sorting bins the categories of the lower bounds of the survivabilities determined by the method described in the previous section. In general, this sorting assigns several options to each bin. To rank the options within the bins Scap uses the costs of the options. Here we have, however, a problem, because the aggregation of costs is not well defined. For instance, it is not clear whether the combination of three actions, each incurring a "small" cost increase is less or more expensive than one action with "large" cost increase. In such cases, the decision maker needs more engineering information. Scap arranges, therefore, the options with equal survivability ranking (equal lower-bound survivability categories) according to the number of cost increases counting savings as negative increases and ignoring the sizes (categories) of the cost changes. The determination of the total costs of the options by aggregating the individual action costs is left to the decision maker.

8. EXAMPLES.

8.1. Three-Element Systems.

To test that part of the program which analyzes the fault trees we constructed two pairs of different but functionally equivalent fault trees and compared the computed results. Scap uses different computing paths for different fault-tree structures, but if the systems are functionally equivalent, then the results should be identical for identical case inputs. In our tests the results were indeed identical.

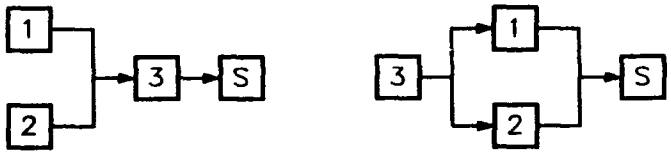

Figure 4. Equivalent Systems 3a and 3b.

The first pair of equivalent systems, called System 3a and 3b, respectively, is shown in Figure 4. The systems consist of three elements labeled 1, 2, and 3. The

fictitious element that represents the system is denoted in the figure by S. The equivalences between systems in this and in the next example pertain to the dependence of the survivabilities of the systems on the survivabilities of Elements 1, 2 and 3. The systems are not equivalent in terms of the interdependencies among the element survivabilities, but these are not of interest for our tests. The description of the System 3a in the data-bank file is illustrated in Appendix D. As a case input for the System 3a (and for the equivalent System 3b) we specified three actions that affected the element survivabilities with respect to 10 of the 16 environments. The actual case-input file is listed in Appendix E. A summary of the present state and the case input is shown in Appendix F, that is part of Scap output. In this example, the input specifies three actions. The number of options (combinations of the actions) is therefore seven. Scap determined that options No. 4, 5, 6 and 7 yield identical lower bounds of system survivabilities and that option No. 4 is the cheapest of these. Table 6 contains a list of these results. It is obvious from the list that action No. 3 alone suffices to produce the the overall survivability level "quite good". Adding other actions to the action No. 3 does not increase the survivability of the system but increases costs. Figure 5 illustrates the increase of the survivability by the best option No. 4 (consisting of action No. 3 only). The figure shows the membership functions of the system survivability before and after hardening. The linguistic interpretation of the result is that the survivability of the system changes by this option from "poor" (about 0.2) to "quite good" (about 0.6). Figure 6 shows for comparison the effect of option No. 1, which consists of the single hardening action No. 1. The linguistic interpretation of the hardening with option No. 1 is that the system survivability changes from "poor" (about 0.2) to "poor to moderate" (about 0.2 to about 0.4).

Table 6. Best Options for Overall Survivability of System 3a.

Name of system: System 3a Present survivability: "poor"			
Options with largest lower bounds of survivability			
Option	Survivability	Actions	Action costs
4	quite good	3	large increase
5	quite good	1 3	medium increase large increase
6	quite good	2 3	small increase large increase
7	quite good	1 2 3	medium increase small increase large increase

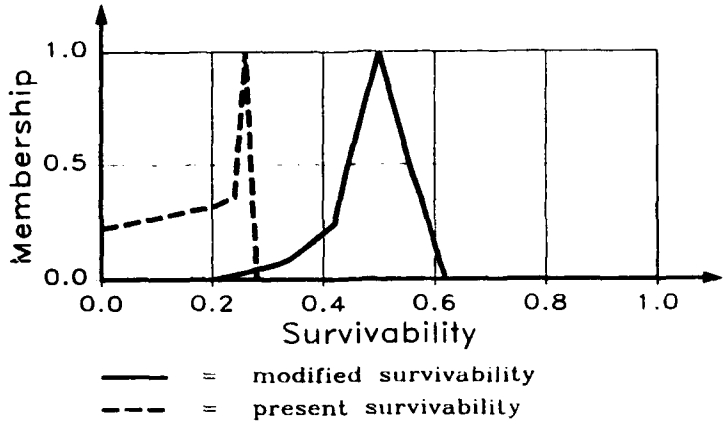

Figure 5. Survivability of System 3a with option No. 4.

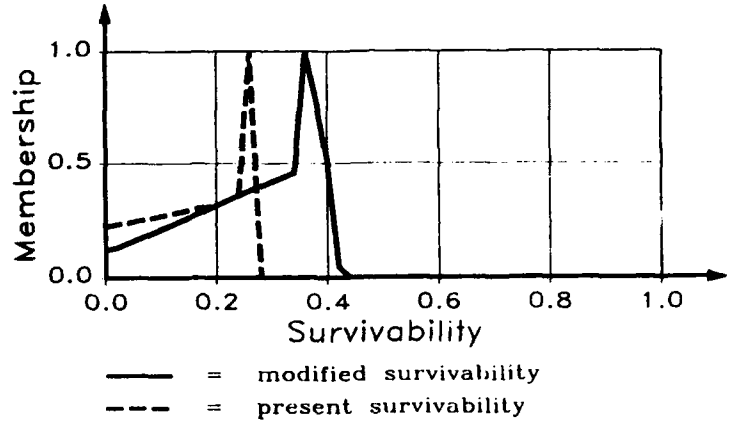

Figure 6. Survivability of System 3a with option No. 1.

A second pair of equivalent systems is shown in Figure 7. For the System 3C (and for the equivalent System 3D) we again proposed three actions resulting in seven options. (These actions were different from those in the first example.) In this example, we instructed Scap to find the best solutions with respect to environments one and two. The results are listed in Table 7 and are shown in Figures 8 and 9. The table shows that the options No. 5, 6 and 7 have equal lower bounds of survivability. The survivability membership curves of options No. 5 and 7 belong to the same category but No. 7 is more expensive. An inspection of the membership functions shows that the system survivabilities of these two options not only belong to the same category but also are identical. Therefore, option No. 7 can be excluded from the list of final choices. This leaves options No. 5 and 6 as contenders for the best place. The final choice must be left to the decision maker who has to estimate the total costs and then decide

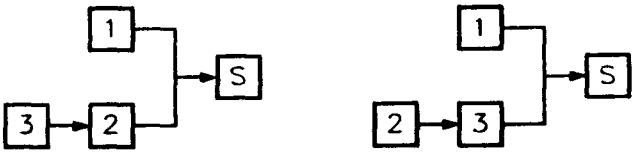

Figure 7. Equivalent Systems 3C and 3D.

Table 7. Best Options for Partial Survivability of System 3C.

Name of system: System 3C Present survivability: 0.26 or "poor" Relevant subset of environments: 1. Over-pressure peak 2. Over-pressure impulse			
Options with largest lower bounds of survivability			
Option	Survivability	Actions	Action costs
5	good to very good	1 3	medium increase large increase
6	good	2 3	small increase large increase
7	good to very good	1 2 3	medium increase small increase large increase

whether the greater survivability with option No. 5 justifies the higher costs.

8.2. One-Element Many-Option System

This example illustrates a situation where the system is considered as one unit and the choice is among hardenings of the system with respect to various parameters. The fault tree consists in this case of a single element that feeds into the system. The relevant survivability was defined as the overall survivability with respect to all environment parameters. The proposed hardening actions and their costs are listed in Table 8. The table contains a list of those environment parameters that are considered for hardening and the amounts of hardening. (The system is assumed to be insensitive with respect to the not-listed parameters. In the data bank, the survivabilities of the element with respect to those parameters are set equal to unity.) The column "Present Survivability" contains the system's survivability with respect to the environment

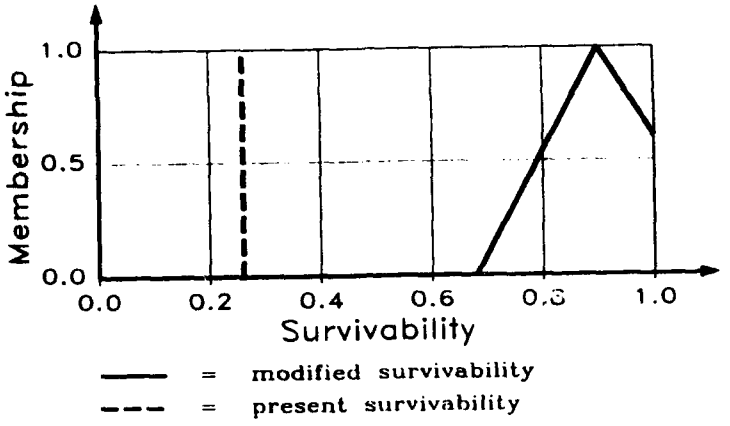

Figure 8. Survivability of System 3C with option No. 5.

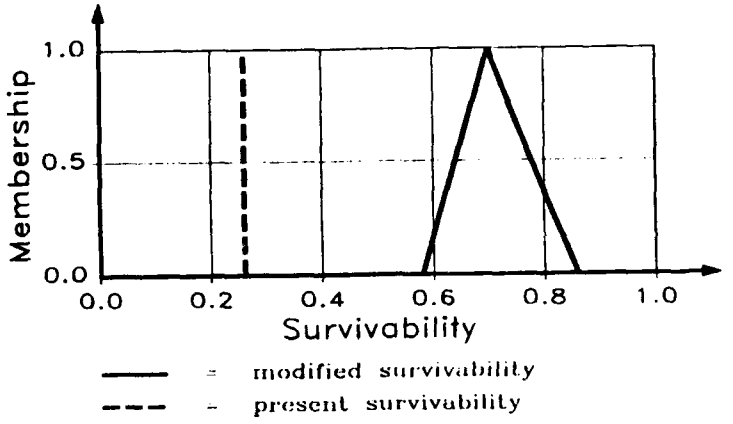

Figure 9. Survivability of System 3C with option No. 6.

parameter in the first column. Note that only two of the entries in this column are linguistic. The other five entries are crisp numbers, and are supplemented with a linguistic interpretation by Scap. The crispness of the present survivability values is recognized by Scap, and the values are handled accordingly. For instance, if the hardening actions were defined by crisp increases of the crisp survivabilities, then the result would be crisp, too. In the present test, we have assumed that all hardening actions are fuzzy and, therefore, the results of the hardening are fuzzy numbers except in cases where a perfect survivability is achieved. The next three columns in Table 8 list the proposed actions, their effects, and costs, respectively. The last column labeled "Result" contains the new system survivabilities (after hardening) with respect to the parameters in the first column. These survivabilities are not input but were computed by Scap. One would like to know which combination of the listed actions is needed for an improvement of the overall survival of the system and how large are the costs for the

best hardening options. Because there are seven proposed actions in this example, the decision maker can choose among 127 different options (combinations of hardening actions).

Table 8. Problem Formulation for a Simple System Hardening.

Name of system: System 4001
Overall present survivability: 0.18 or "poor"

Param.	Present Surviv.	Action	Hardening	Costs	Result
2	0.18 (poor)	1	large med.	medium	very good
4	good	2	medium	small	1.0
5	0.56 (quite good)	3	small	large	very good
8	0.42 (moderate)	4	medium	medium	very good
11	0.82 (good)	5	small med.	medium	1.0
12	0.72 (good)	6	medium	small	1.0
13	quite good	7	small	large	very good

Scap determined that the greatest overall survivability level ("very good") can be achieved only by the option No. 127 that contains all seven actions. By checking the last column in Table 8 one can easily verify this result. Figure 10 illustrates the achieved overall survivability. Having this result, the decision maker may investigate if some of the hardening actions could be replaced by cheaper and less effective ones without impairing the overall survivability. Candidates for such modifications are the hardenings with respect to the parameters No. 4, 11, and 12, that produce perfect survivabilities. In a further analysis, the decision maker might also consider those environment parameters (not listed in Table 8) that do not affect the present system survivability. Since the overall survivability level that can be achieved is only "very good" (and not a perfect 1.0) then possibly some savings could be realized by softening the system with respect to those parameters. Finally, if one is content with a "good" survivability, a simpler option that involves only the actions 1, 3, 4, and 7 might be selected.

9. SUMMARY AND CONCLUSIONS.

This report presents a description of a pilot computer program Scap that computes the survivability of weapon systems in terms of proposed changes (hardening or softening) of the survivabilities of system elements. Some of the information about prospective changes is allowed to be vague, and fuzzy-set theory is used for the representation and handling of such information. The program was exercised for several test cases and found to perform as expected. Experiments with the program show that the same general structure can be used as a basis for a full scale utility program. Such a program should be useful as a support for cost-effectiveness analysis of nuclear

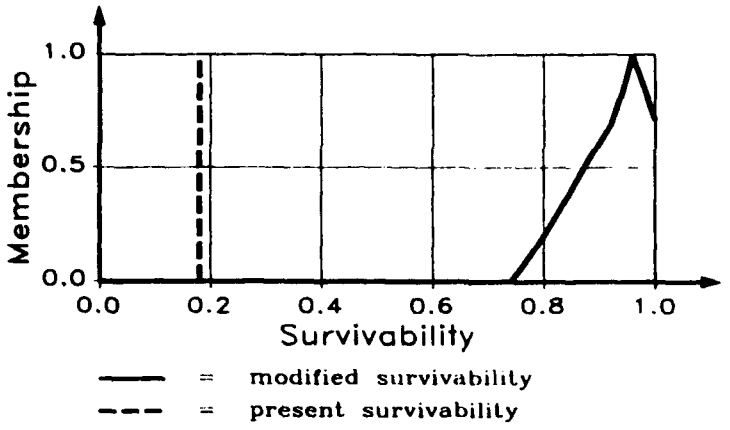

Figure 10. Overall survivability of System 4001 with option No. 127.

survivability of weapon systems. The survivability membership functions were internally represented in the pilot program by arrays with 51 elements. We found this representation too coarse and suggest a representation by 101 elements in a final utility program. Some restrictions on the number of system elements and on the number of proposed actions might be necessary if the final program is intended for a personal computer with restricted memory. The computing times on a minicomputer or a main frame computer were a fraction of one minute. We consider this to be acceptable for the intended applications.

INTENTIONALLY LEFT BLANK

10. REFERENCES.

Bortolan, G. and R. Degani. "A review of some methods for ranking fuzzy subsets." Fuzzy Sets and Systems **15**, pp. 1-19, 1985.

Choobineh, F. and Huishen Li. "An index for ordering fuzzy numbers." Fuzzy Sets and Systems **54**, pp. 287-294, 1993.

Dubois, Didier and Henri Prade. "On the ranking of ill-known values in possibility theory." Fuzzy Sets and Systems **43**, pp. 311-317, 1991.

Kaufmann, Arnold and Madan M. Gupta. Introduction to Fuzzy Arithmetic. Van Nostrand Reinhold Company, New York, NY, 1985.

Kim, Kuk and Kyung S. Park. "Ranking Fuzzy Numbers with Index of Optimism." Fuzzy Sets and Systems **35**, pp. 143-150, 1990.

Klir, George J. and Tina A. Folger. Fuzzy Sets, Uncertainty and Information, Prentice Hall, Englewood Cliffs, NJ, 1988.

Zadeh, Lotfi A. "Fuzzy Sets." Information and Control **8**, pp. 338-353, 1965.

Zimmermann, Hans-Jürgen. Fuzzy set theory – and its applications, 2nd edition, Kluwer Academic Publishers, Boston, MA, 1991.

INTENTIONALLY LEFT BLANK

Appendix A.

COMPUTATION OF BASE–LINE SURVIVABILITIES OF ELEMENTS.

INTENTIONALLY LEFT BLANK

The survivability s of a system element with respect to an environment parameter p is described by a fuzzy survivability function $s(p)$. Because there are 16 environment parameters, 16 such functions are needed to completely characterize the survivability properties of an element. Each function is defined in the data bank by six curve parameters that are the abscissas of the points t_1, t_2, t_3, b_1, b_2, and b_3 shown in Figure A1. Let the given value of the environment parameter be a fuzzy number $\widetilde{P}$ with the membership function $\mu_P(p)$. We want to compute the corresponding value $\widetilde{S} = s(\widetilde{P})$ of the element survivability. We calculate the membership function $\mu_S(s)$ of $\widetilde{S}$ by interpolation in the given function $s(p)$, that is, by intersecting the membership function $\mu_{s(p)}(s,p)$ of the survivability curve $s(p)$ with the parameter-membership function $\mu_P(p)$ of $\widetilde{P}$ and projecting the intersection onto the survivability axis. This operation is formally expressed by

$$\mu_S(s) = \sup_p \min \{ \mu_{s(p)}(s,p) , \mu_P(p) \} .$$

To carry out this calculation, we need the values of the function $\mu_{s(p)}(s,p)$, that is, the membership value of the fuzzy survivability function $s(p)$ for given values of p and s. We obtain these membership values by interpolation between the curve-parameter points as indicated by the level lines in Figure A1. We now list the formulas for the interpolation.

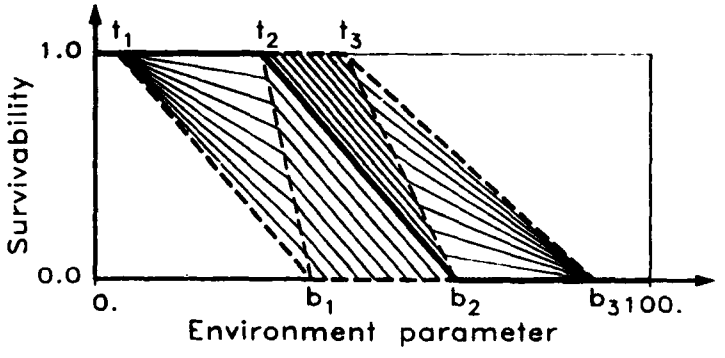

Figure A1. Fuzzy element-survivability curve with level lines.

The interpolation algorithm consists of the following steps. First, we determine for the given s the abscissas p_A, p_B, p_C, p_D, and p_E of the points A, B, C, D, and E in Figure A2. These points divide the p-axis into six intervals. Next, we determine to which interval the given p belongs and use a corresponding interpolation formula to compute the membership value $\mu_{s(p)}(s,p)$. The formulas for the five abscissas are

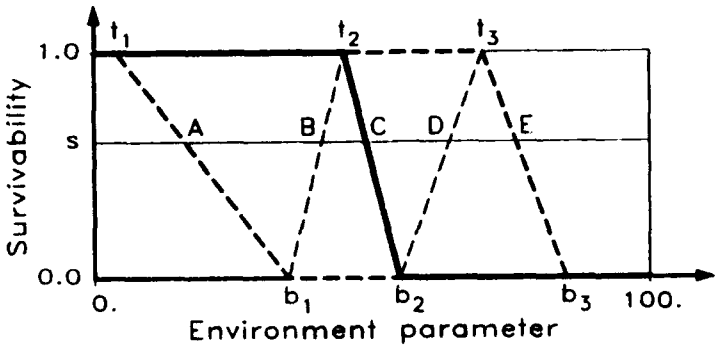

Figure A2. Interpolation intervals.

$$p_A = b_1 + s\,(t_1 - b_1)\ ,$$
$$p_B = b_1 + s\,(t_2 - b_1)\ ,$$
$$p_C = b_2 + s\,(t_2 - b_2)\ ,$$
$$p_D = b_2 + s\,(t_3 - b_2)\ ,$$
$$p_E = b_3 + s\,(t_3 - b_3)\ .$$

The interpolation formulas are as follows:

$p \le p_A$:
$$\mu_{s(p)}(s,p) = 1 \quad \text{if} \quad s = 1\ ,$$
$$\mu_{s(p)}(s,p) = 0 \quad \text{if} \quad s < 1\ .$$

$p_A < p < p_B$:
$$\mu_{s(p)}(s,p) = \frac{p - p_A}{p - p_A + (1-s)\,(t_2 - t_1)}\ .$$

$p_B \le p < p_C$:
$$\mu_{s(p)}(s,p) = s + (1-s)\,\frac{p - p_B}{p_C - p_B} = s + \frac{p - p_B}{b_2 - b_1}\ .$$

$p_C \le p \le p_D$:
$$\mu_{s(p)}(s,p) = 1 - s + s\,\frac{p_D - p}{p_D - p_C} = 1 - s + \frac{p_D - p}{t_3 - t_2}\ .$$

$p_D < p < p_E$:
$$\mu_{s(p)}(s,p) = \frac{p_E - p}{p_E - p + s\,(b_3 - b_2)}\ .$$

$p_E \le p$:
$$\mu_{s(p)}(s,p) = 0 \quad \text{if} \quad s > 0\ ,$$
$$\mu_{s(p)}(s,p) = 1 \quad \text{if} \quad s = 0\ .$$

The interpolation formulas for the intervals $[p_B, p_C]$ and $[p_C, p_D]$ are obvious from Figure A1.

The interpolation formulas for the intervals $[p_A, p_B]$ and $[p_D, p_E]$ are derived as follows. Let $\vec{Q}$, $\vec{R}$, $\vec{S}$, and $\vec{T}$ define two straight lines in the plane, as shown in Figure A3. Then the equations for the lines are in parameter form

$$\vec{L}_1(\sigma) = \vec{S} + (\vec{T} - \vec{S})\,\sigma$$

and

$$\vec{L}_2(\tau) = \vec{Q} + (\vec{R} - \vec{Q})\,\tau$$

where σ and τ are scalar parameters. We have indicated in Figure A3 the points where the parameters have the values zero and unity, respectively. The intersection of the two lines is obtained by computing the values of the parameters σ and τ from the intersection equation $\vec{L}_1(\sigma) = \vec{L}_2(\tau)$, or

$$(\vec{T} - \vec{S})\,\sigma - (\vec{R} - \vec{Q})\,\tau = \vec{Q} - \vec{S} \quad .$$

Let the components of $\vec{T}$ be T_1 and T_2, and corresponding for the other vectors. In terms of these components the intersection equation is

$$\begin{aligned} (T_1 - S_1)\,\sigma - (R_1 - Q_1)\,\tau &= Q_1 - S_1 \quad , \\ (T_2 - S_2)\,\sigma - (R_2 - Q_2)\,\tau &= Q_2 - S_2 \quad . \end{aligned}$$

Solving this system of equations for σ, we obtain

$$\sigma = \frac{(R_2 - Q_2)\,(Q_1 - S_1) - (R_1 - Q_1)\,(Q_2 - S_2)}{(R_2 - Q_2)\,(T_1 - S_1) - (R_1 - Q_1)\,(T_2 - S_2)} \quad .$$

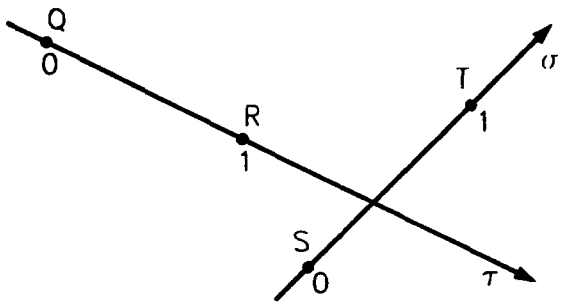

Figure A3. Derivation of interpolation formulas.

To establish the interpolation formula for the interval $[p_A, p_B]$, we make in this expression the following substitutions that can be read from Figure A1: $\vec{Q}^T = (t_1, 1)$, $\vec{R}^T = (p, s)$, $\vec{S}^T = (b_1, 0)$, and $\vec{T}^T = (t_2, 1)$. The membership value μ and the parameter σ increase linearly between the points $\vec{S}$ and $\vec{T}$ from zero to unity, and the line through $\vec{Q}$ and $\vec{R}$ is a level line in Figure A1. Therefore, the membership value $\mu_{s(p)}(s,p)$ at the point $\vec{R}$ is equal to σ at the intersection point.

The interpolation formula for the interval $[p_D, p_E]$ is obtained by substituting in the above expression $\vec{Q}^T = (b_3, 0)$, $\vec{R}^T = (p, s)$, $\vec{S}^T = (t_3, 1)$, and $\vec{T}^T = (b_2, 0)$.

INTENTIONALLY LEFT BLANK

Appendix B.

LOGICAL COMBINATIONS OF SURVIVABILITIES.

INTENTIONALLY LEFT BLANK

We provide in this appendix formulas for logical combinations of element survivabilities. We consider first a conjunctive combination. Let the system e_W consist of two conjunctive elements e_A and e_B, that is, the system survives if e_A and e_B survive. Figure B1 shows this relation in the form of a fault tree and in the form of a logical gate. The survivability of the system e_W should be the same in either representation if the survivabilities of the elements are the same. Let A and B be the crisp survivabilities of the elements e_A and e_B, respectively. Then the survivability W of the system e_W is

$$W = \min \{ A , B \} .$$

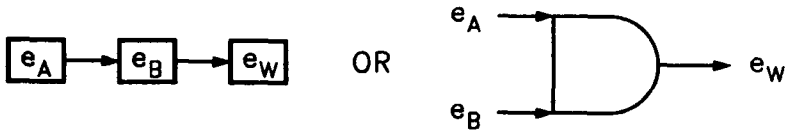

Figure B1. Conjunctive system.

Now let the element survivabilities be given by fuzzy numbers $\tilde{A}$ and $\tilde{B}$ and let the corresponding possibility distributions (membership functions) of the survivability s be

$$\pi_A(s) = \pi_{s \in \tilde{A}}(s) \qquad \text{and} \qquad \pi_B(s) = \pi_{s \in \tilde{B}}(s) .$$

Then the membership function or possibility distribution $\pi_W(s)$ of the survivability $\tilde{W}$ of the system is computed by

$$\pi_W(s) = \pi_{s \in \min\{\tilde{A},\tilde{B}\}}(s) = \pi_{[\, s < \tilde{A} \wedge \tilde{B} \,] \wedge [\, s > \tilde{A} \vee \tilde{B} \,]} =$$
$$= \min \left\{ \pi_{s < \tilde{A} \wedge \tilde{B}} , \pi_{s > \tilde{A} \vee \tilde{B}} \right\} =$$
$$= \min \left\{ \min \left\{ \pi_{s < \tilde{A}} , \pi_{s < \tilde{B}} \right\} , \max \left\{ \pi_{s > \tilde{A}} , \pi_{s > \tilde{B}} \right\} \right\}$$

where $\wedge$ is the logical "and" operator and $\vee$ is the logical "or" operator. The last line shows how to compute the membership function $\pi_W(s)$ by min and max operations if the functions $\pi_{s < \tilde{A}}(s)$, $\pi_{s < \tilde{B}}(s)$, $\pi_{s > \tilde{A}}(s)$, and $\pi_{s > \tilde{B}}(s)$ are known. The formula selects the leftmost distribution if $\pi_A(s)$ and $\pi_B(s)$ are disjoint and in cases where the distributions are triangular or trapezoidal with equal slopes. An example of a case with intersecting distributions and unequal slopes is shown in Figure B2. The practical computation of the distributions $\pi_{s < \tilde{A}}$, $\pi_{s < \tilde{B}}$, $\pi_{s > \tilde{A}}$ and $\pi_{s > \tilde{B}}$ that enter the formula for $\pi_W(s)$ is done in Scap as follows.

The survivability is restricted to the interval [0,1]. One can, therefore, represent all survivability membership functions by arrays with a fixed number of elements, each

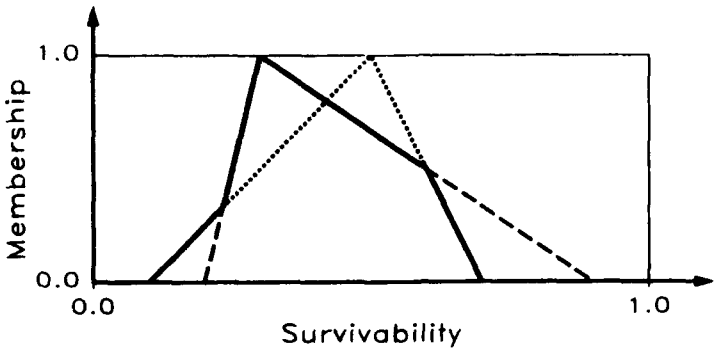

Fig .e B2. Conjunctive combination of survivabilities.

element denoting the membership value of a fixed survivability. In Scap, the array length was chosen to be 51 so that the values of the array elements denote the memberships of the survivability values 0.0, 0.02, 0.04, ..., 0.98, and 1.0. (Based on our numerical experiments we believe that 101 or more elements would be more appropriate for the representation of arbitrary membership functions.) This representation allows to approximate continuous membership functions by discrete fuzzy sets with a fixed number of elements. Membership values between the nodes defined by the arrays are obtained in Scap by linear interpolation. Having discrete sets with a fixed number of elements greatly simplifies the computation of $\pi_W(s)$. For example, the calculation of the possibility distribution of $s > \widetilde{A}$ that is necessary to calculate $\pi_W(s)$ is done as follows. Let $A(j)$, $j = 1,...,51$ be the array representing the membership function $\pi_A(s)$ of $\widetilde{A}$, and $G(j)$, $j = 1,...,51$ be the array representing the wanted membership function $\pi_{s > \widetilde{A}}(s)$. Then $G(j)$ can be computed by the following simple loop:

```
      G(1) = A(1)
      do 3,  j = 2,51
      G(j) = max ( G(j-1), A(j) )
3     continue
```

The other three distributions, $\pi_{s < \widetilde{A}}$, $\pi_{s < \widetilde{B}}$, and $\pi_{s > \widetilde{B}}$, can be computed by corresponding equally simple loops.

Next, we consider a system e_W that consists of two disjunctively connected elements e_A and e_B (the system survives if at least one of the elements survives). The corresponding fault-tree and logical-gate representations are shown in Figure B3. Let the element survivabilities be A and B. Then the survivability W of the system is given by the formula

$$W = \max \{ A , B \} .$$

If the survivabilities of the elements are fuzzy numbers $\widetilde{A}$ and $\widetilde{B}$ with the possibility distributions $\pi_A(s)$ and $\pi_B(s)$, respectively, then the aggregation formula is

$$\pi_W(s) = \pi_{s \in \max\{\tilde{A},\tilde{B}\}}(s) = \pi_{[s > \tilde{A} \wedge \tilde{B}] \wedge [s < \tilde{A} \vee \tilde{B}]} =$$

$$= \min\left\{ \pi_{s > \tilde{A} \wedge \tilde{B}}, \pi_{s < \tilde{A} \vee \tilde{B}} \right\} =$$

$$= \min\left\{ \min\left\{\pi_{s > \tilde{A}}, \pi_{s > \tilde{B}}\right\}, \max\left\{\pi_{s < \tilde{A}}, \pi_{s < \tilde{B}}\right\} \right\} .$$

This formula selects the rightmost distribution if $\pi_A(s)$ and $\pi_B(s)$ are disjoint. A more interesting example is shown in Figure B4.

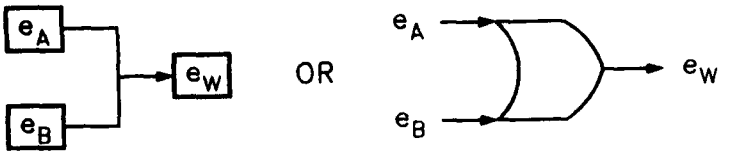

Figure B3. Disjunctive system.

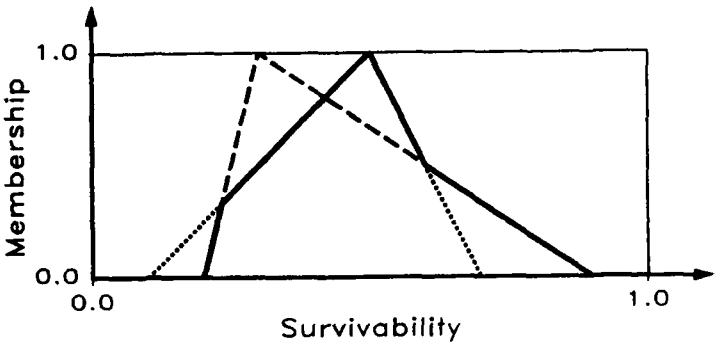

Figure B4. Disjunctive combination of survivabilities.

INTENTIONALLY LEFT BLANK

Appendix C.

LIST OF THE PROGRAM SCAP

INTENTIONALLY LEFT BLANK

```
      program scap
*Main program for survivability cost analysis.  21 August 1992
*
      dimension envir(3,16)
      integer cstact(7),hrdact(13,16,7),inqui(3),inqsub(16),
     a mempl1(2,10),mempl2(10),mempl3(10)
*
      character sysid*30,itemid(13)*30
      dimension surbas(6,16,13)
      integer conseq(13,13),antand(13,13),antor(13,13)
*
      dimension surcur(51,16,13),surexi(51,16,13),hasurc(51,16,13)
*
      integer kasact(7)
*
      call rdcurr(envir,sysid,nract,cstact,hrdact,inqui,inqsub,
     a np1,mempl1,np2,mempl2,np3,mempl3)
*  read from file 'input-scap'  the following current values:
*  envir(3,16) - environment: (1,c,r) for 16 environment parameters
*  sysid - system AID  (alphanumeric identification)
*  nract - number of actions to be investigated  (<= 7)
*  cstact(7) - costs of up to 7 proposed actions
*  hrdact(13,16,7)  - hardening categories of the actions (13 elements,
*              16 environments, 7 actions)
*  inqui(3)  -  specification of questions asked (3 types of inquiry)
*  inqsub(16) -  subset (list) of environment parameters that is to be analyzed
*               in option inqui(2).ne.0
*  np1,np2,np3 -  numbers of membership plots for each inquiry
*  mempl1, mempl2, mempl3 - indicators of options for which membership
*                 functions should be plotted
*
      call rddatb(sysid,nritms,itemid,conseq,antand,antor,surbas,nbadd)
*  read from data base file 'datbank-scap' for the system "sysid" the following
*  nritms  -  number of items (elements) in the system
*  itemid(13) - item AIDs
*  conseq(13,13)  -  list of consequence elements for each element in the system
*  antand(13,13)  -  conjunctive antecedent elements for each element
*  antor(13,13)  -  disjunctive antecedent elements for each element
*  surbas(6,16,13)  -  basic (unaltered) survivability functions (6 nodes)
*            for  16 environments and 13 elements
*  nbadd = 99 if the specified system (sysid) is not in the data base
*
      if(nbadd.ne.0) then
      open(unit=1,file='scap-message')
      rewind(unit=1)
      write(1,11) nbadd
   11 format('Stop because data base reader rddatb returns with'
     a ,' nbadd='i2':'/' The system cannot be found in the data base.')
      close(unit=1)
      stop
      endif
*
      call cursur(envir,nritms,surbas,surcur)
```

```
* compute current survivability memberships "surcur" for all elements
* surcur(51,16,13) - survivability memb. functions for 16 env. & 13 elements
    call curexs(nritms,conseq,antand,antor,surcur,surexi,nbadc)
*  compute current exit survivability "surexi"  of the unaltered system
* surexi(51,16,13) - exit surv. memberships for 16 envir. and 13 elements
* surexi(51,16,nritms+1) - exit survivability of the system for 16 envir.
    if(nbadc.ne.0) then
    open(unit=1,file='scap-message')
    rewind(unit=1)
    write(1,13) nbadc
   13 format('Stop because curexs returns with'
    a ,' nbadc='i2)
    close(unit=1)
    stop
    endif
*
    call inprnt(envir,sysid,nract,cstact,hrdact,
   a  nritms,itemid,surcur,surexi,inqui,inqsub)
*  Write comprehensive input summary on  'stin-scap'
*
    kase=0
    call stor(sysid,nritms,kase,kasact,cstact,hrdact,surexi,
   a inqui,inqsub,np1,mempl1,np2,mempl2,np3,mempl3)
*  Interpret and store in output files  present (unaltered) survivability
*  membership function  "surexi"  (kase=0 indicates unaltered "option"))
*
    kasmax=2**nract-1
*  Total number of all combinations of proposed actions
*
   15 kase=kase+1
* Start here a loop over  kase=1,kasmax  combinations of actions
*
    call comb(nract,kase,kasact)
* "comp"  computes a combination of actions and stores it in kasact
* kasact(7) -  ones indicate that the corresponding action is taken
*
    call nusurv(nritms,kasact,hrdact,surcur,hasurc)
*  Compute modified survivabilities "hasurc"  due to hardening  hrdact
*  for the combination "kasact" of hardening actions
*  hasurc(51,16,13) - hardened surviv. memberships for 16 envir. of 13 elements
*
    call curexs(nritms,conseq,antand,antor,hasurc,surexi,nbadc)
*  compute new (hardened) exit survivability "surexi" of the system
*  using modified survivability curves "hasurc"
*  surexi(51,16,13) - exit surv. memberships for 16 envir. and 13 elements
*
    call stor(sysid,nritms,kase,kasact,cstact,hrdact,surexi,
   a inqui,inqsub,np1,mempl1,np2,mempl2,np3,mempl3)
*  Interpret and store in output files the new (modified) survivability
*  membership function "surexi" for this "kase"
*   (also store other information given by the arguments)
*
    if(kase.lt.kasmax) goto 15
*  Branch for next "kase" if all combinations of actions are not exhausted
*
    if(inqui(1).ne.0) call anal1
```

```
      if(inqui(2).ne.0) call anal2
      if(inqui(3).ne.0) call anal3
*  Call analysis programs
*
      stop
      end
*
      subroutine comb(nract,kase,kasact)
*  Indicate which current option to take from all combinations of actions.
*  nract = number of elements that are changed
*  kase = ID-nr of the combination (option) of changed elements
      dimension kasact(7)
* On return, ones in kasact will indicate which of up to 7 elements are changed
*
      do 12 kb=1,nract
      kasact(kb)=0
   12 continue
*
      kk=kase
      kd=2**nract
      do 35 kr=1,nract
      kd=kd/2
* Represent kase in binary form and store the binary components in kasact
       if(kk.ge.kd) then
       kasact(nract+1-kr)=1
       kk=kk-kd
       endif
   35 continue
      return
      end
*
      subroutine rdcurr(envir,sysid,nract,cstact,hrdact,inqui,inqsub,
     a np1,mempl1,np2,mempl2,np3,mempl3)
*  Read from the input file 'input-scap' (unit 3) current environment,
*  proposed actions and specifications of inquiries.
*  24 August 1992
*
*  envir(3,16) - current values (low,center,high) of 13 environment parameters
*  sysid  -  system AID (alphanumeric identification)
*  nract  -  number of "actions" to be investigated (le.7)
*  cstact(7) -  cost categories [-3,3] associated with the actions
*  hrdact(13,16,7) - hardening categories [-5,5] for 13 elements,
*            16 environments and 7 actions
*  inqui(3) - type of inquiry: (1)=1 - give results for each environment
*                (2)=1 - give combined results for a subset
*                (3)=1 - give combined result for all environments
*  inqsub(16) - subset list of environment parameters for inquiry "(2)=1"
*  np1,np2,np3 -  numbers of membership plots for each inquiry
*  mempl1(2,10), mempl2(10), mempl3(10) - indicators of options for which
*           membership functions should be plotted
*
      dimension envir(3,16)
      integer cstact(7),hrdact(13,16,7),inqui(3),inqsub(16),invec(16),
     a mempl1(2,10),mempl2(10),mempl3(10)
      character sysid*30,text*30
*
```

```
      do 8 ka=1,7
      cstact(ka)=0
      do 7 kb=1,16
      do 6 kc=1,13
      hrdact(kc,kb,ka)=0
    6 continue
    7 continue
    8 continue
      do 10 ka=1,10
      mempl1(1,ka)=0
      mempl1(2,ka)=0
      mempl2(ka)=0
      mempl3(ka)=0
   10 continue
*
      open(unit=3,file='input-scap')
      rewind(unit=3)
*
      read(3,12) text
   12 format(a30)
*
      read(3,12) text
* Read environment parameter Nr, and low bound, center, high bound of parameter
      do 22 kl=1,16
      read(3,*) ke,(envir(j,ke),j=1,3)
   22 continue
*
      read(3,12) text
* Read system's AID  (alphanumeric identification)
      read(3,12) sysid
*
      read(3,12) text
* Read number of actions to be analyzed
      read(3,*) nract
*
      do 74 kact=1,nract
*
      read(3,12) text
* Read Action ID-nr, nr of elem. changed, change of costs (integer in [-3,3])
      read(3,*) idact,nritch,cstact(idact)
      cstact(idact)=max(-3,min(3,cstact(idact)))
*
      read(3,12) text
* Read  Element ID-nr and 16 hardness changes (integer in [-5,5])
      do 24 nri=1,nritch
      read(3,*) iditem,(hrdact(iditem,j,idact),j=1,16)
   24 continue
      do 25 j=1,16
      hrdact(iditem,j,idact)=max(-6,min(6,hrdact(iditem,j,idact)))
   25 continue
*
   74 continue
*
      read(3,12) text
* Read inquiries' specification: Each [0,1], Subset of 16 [0,16], All 16 [0,1].
      read(3,*) (inqui(j),j=1,3)
```

```
      inqui(2)=max(0,min(16,inqui(2)))
*
      read(3,12) text
* Read the list of environment parameters that are in the subset
      do 84 ka=1,16
      inqsub(ka)=0
   84 continue
      jtop=max(1,inqui(2))
      read(3,*) (invec(j),j=1,jtop)
      if(inqui(2).gt.0) then
      do 91 ka=1,jtop
      inqsub(invec(ka))=1
* inqsub contains "1" in places corresponding to subset of environment
   91 continue
      endif
*
      np1=0
      np2=0
      np3=0
      read(3,12) text
*  Read the list of options for which membership functions should be plotted
      read(3,*) nplots
      if(nplots.le.0) goto 97
*
      read(3,12) text
      nplots=min(10,nplots)
      do 95 ka=1,nplots
      read(3,*) kq,kc,kp
      if(kq.eq.1) then
       np1=np1+1
       mempl1(1,np1)=kc
       mempl1(2,np1)=kp
      endif
      if(kq.eq.2) then
         np2=np2+1
         mempl2(np2)=kc
      endif
      if(kq.eq.3) then
          np3=np3+1
          mempl3(np3)=kc
      endif
   95 continue
*
   97 close(unit=3)
      return
      end
*
      subroutine rddatb(sysid,nritms,itemid,conseq,antand,antor,surbas,
     a nbadd)
* Read from data bank file 'datbank-scap' (unit 4) specifications of
* this system
*  24 August 1992
*
*  sysid - system's AID (Alphanumeric ID)
*  nritms - number of elements in this system (.le.12, elem. nr.13 is "system")
*  itemid(13) - element AIDs
```

```
*  conseq(i,j) - consequences "i" of element "j"
*  antand(i,j) - conjunctive antecedents "i" of element "j"
*  antor(i,j)  - disjunctive antecedents "i" of element "j"
*  surbas(6,16,13) - survivability functions (defined by 6 parameters)
*         for 16 environments and 13 elements
*  nbadd - error return with nbadd=99 if the system is not in the data base
*
     character sysid*30,itemid(13)*30
     dimension surbas(6,16,13)
     integer conseq(13,13),antand(13,13),antor(13,13)
     character text*30
*
     nbadd=0
     open(unit=4,file='datbank-scap')
     rewind(unit=4)
*
     read(4,12) text
   12 format(a30)
     read(4,12) text
*
   14 read(4,12,end=15) text
     if(text.ne.sysid) goto 14
* Find the file for this system in the data bank
     goto 17
   15 nbadd=99
     return
* Return because the system file cannot be found in the data base
*
   17 read(4,12) text
     read(4,*) nritms
* This is the number of items (elements) in this system
*
     kksys=nritms+1
     do 85 kk=1,kksys
* Loop over all elements of this system
*  Element with the number "kksys" is the "system" itself
*
     read(4,12) text
* Read element ID-nr. and element AID (alphanumeric identification)
     read(4,21) kit, itemid(kit)
   21 format(i2,a30)
*
     do 34 ka=1,13
     conseq(ka,kit)=888
     antand(ka,kit)=888
     antor(ka,kit)=888
   34 continue
* Clear storage (set all consequences and antecedents equal dummy element)
*
     read(4,12) text
*  Read number of consequences and their ID-nrs.
     read(4,*) icnr,(conseq(j,kit),j=1,icnr)
*
     read(4,12) text
*  Read number of "AND" antecedents and their ID-nrs
     read(4,*) iand,(antand(j,kit),j=1,iand)
```

```
*
      read(4,12) text
*  Read number of "OR" antecedents and their ID-nrs
      read(4,*) ior,(antor(j,kit),j=1,ior)
*
      read(4,12) text
* Read survivabilities w/respect of the 16 environment parameters
      do 55 ka=1,16
      read(4,*) nrpar,(surbas(j,nrpar,kit),j=1,6)
* Each survivability curve is specified by 6 numbers: t1,t2,t1 and b1,b2,b3
   55 continue
*
   85 continue
* End of loop over all elements of this system
*
      close(unit=4)
      return
      end
*
      subroutine cursur(envir,nritms,surbas,surcur)
* Compute for all elements current element-survivability membership functions
* m(s) by conjunctive combination of the current environment parameter
* membership function m(p) with the element-survivability curve m(s,p).
*  24 August 1992
*
* envir(3,16) - current environment (low,center,high) parameter values
*             for 16 environments
* nritms - number of elements in the system (<13) (Nr. 13 to store "system")
* surbas(6,16,13) - data base survivability curves (6 parameters) for
*             16 environment parameters and 13 items (elements)
*
* The following will be computed
* surcur(51,16,13) - current survivability membership curves (51 entries)
*             for 16 environment parameters and 13 items (elements)
*
      dimension envir(3,16),surbas(6,16,13),surcur(51,16,13)
*
      do 115 item=1,nritms+1
* Loop over all elements including the "system" that has the ID-number nritms+1
*
      do 105 ke=1,16
* Loop over all environment parameters
*
      ksspec=0
      do 95 ks=1,51
*  Loop over 51  survivability values (curve entries)
      if(ks.eq.ksspec) goto 95
*  ksspec is the entry with  membership value 1, computed in first pass (ks-1)
      s=float(ks-1)/50.
      surcur(ks,ke,item)=0.
*
      kpmin=100
      kpmax=100
        if(envir(1,ke).lt.envir(2,ke)) kpmin=1
        if(envir(2,ke).lt.envir(3,ke)) kpmax=199
* If support .gt. 0  then chose 199 points "p" within the support of  m(p)
```

```
* (excluding the boundaries of the support)
* and compute the corresponding (positive) membership values of "p"
    do 85 kp=kpmin,kpmax
* Loop over current environment parameter support.  At the core have kp=100
*
    if(kpmin.eq.kpmax) then
    p=envir(2,ke)
    pmem=1.
* In this case have crisp environment parameter p
*  pmem  is the membership of the present environment parameter
    else if(kp.lt.100) then
    p=envir(1,ke)+(envir(2,ke)-envir(1,ke))*float(kp)/100.
    pmem=float(kp)/100.
* left hand part of parameter membership curve
    else
    p=envir(2,ke)+(envir(3,ke)-envir(2,ke))*float(kp-100)/100.
    pmem=float(200-kp)/100.
*  right hand part of parameter membership curve
    endif
*
*  If ks=1 and  pmem=1  (kp=100)  then find the intersection with smem=1
*   and set surcur(ksspec,..,..)=1  at the closest entry "ksspec"
    if(ksspec.gt.0) goto 55
*  Compute the core coordinate ksspec only once per environment ke
    if(ks.gt.1.or.kp.ne.100) goto 55
    if(p.le.surbas(2,ke,item)) ksspec=51
    if(p.ge.surbas(5,ke,item)) ksspec=1
     if(p.gt.surbas(2,ke,item).and.p.lt.surbas(5,ke,item)) then
     spc=(surbas(5,ke,item)-p)/(surbas(5,ke,item)-surbas(2,ke,item))
*  Intersect "p"  with the core of the survivability curve
     ksspec=1+nint(spc*50.)
     endif
    surcur(ksspec,ke,item)=1.
    if(ks.eq.ksspec) goto 95
*  Go to the end of the "ks" loop
*
*  Next compute interpolation intervals of survivability curve for given "s"
  55 pa=surbas(4,ke,item)+s*(surbas(1,ke,item)-surbas(4,ke,item))
    pb=surbas(4,ke,item)+s*(surbas(2,ke,item)-surbas(4,ke,item))
    pc=surbas(5,ke,item)+s*(surbas(2,ke,item)-surbas(5,ke,item))
    pd=surbas(5,ke,item)+s*(surbas(3,ke,item)-surbas(5,ke,item))
    pe=surbas(6,ke,item)+s*(surbas(3,ke,item)-surbas(6,ke,item))
*
    if(pa.ge.pe.and.envir(1,ke).ge.envir(3,ke)) then
*  In this case have crisp s-curve and crisp environment
*   The intersection has been computed, see ksspec above.
     spmem=0.
     goto 74
    endif
*
* The next six "if" treat the case with crisp  s-curve and fuzzy envir  p
    if(pa.ge.pc) then
     if(envir(1,ke).lt.pa.and.pa.lt.envir(2,ke).and.p.lt.pa) then
     spmem=(pa-envir(1,ke))/(envir(2,ke)-envir(1,ke))
     goto 74
     endif
```

```
      if(envir(2,ke).lt.pa.and.pa.lt.envir(3,ke).and.p.lt.pa) then
      spmem=(envir(3,ke)-pa)/(envir(3,ke)-envir(2,ke))
      goto 74
      endif
     endif
     if(pc.ge.pe) then
      if(envir(1,ke).lt.pa.and.pa.lt.envir(2,ke).and.p.gt.pa) then
      spmem=(pa-envir(1,ke))/(envir(2,ke)-envir(1,ke))
      goto 74
      endif
      if(envir(2,ke).lt.pa.and.pa.lt.envir(3,ke).and.p.gt.pa) then
      spmem=(envir(3,ke)-pa)/(envir(3,ke)-envir(2,ke))
      goto 74
      endif
     endif
*
* Next is the general case with fuzzy environment p and fuzzy s-curve
     if(p.le.pa) then
      if(s.ge.1.) smem=1.
      if(s.lt.1.) smem=0.
     else if(p.le.pb) then
       smem=(p-pa)/(pb-p+surbas(2,ke,item)-surbas(1,ke,item))
     else if(p.le.pc) then
       smem=s + (p-pb)/(surbas(5,ke,item)-surbas(4,ke,item))
     else if(p.le.pd) then
       smem=1.-s+(pd-p)/(surbas(3,ke,item)-surbas(2,ke,item))
     else if(p.lt.pe) then
       smem=(pe-p)/(pd-p+surbas(6,ke,item)-surbas(5,ke,item))
     else if(p.ge.pe) then
        if(s.le.0.) smem=1.
        if(s.gt.0.) smem=0.
     endif
* "smem" is the membership of the survivability curve for (p,s)
* "pmem" is the membership of the environment parameter value p
     spmem=min(pmem,smem)
*
  74 surcur(ks,ke,item)= max(spmem,surcur(ks,ke,item))
* maximum over the support (kp=1,..,199) of current environment parameter
*
  85 continue
* end of kp  loop over up to 199 p-values
  95 continue
* end of ks  loop over 51 s-values
*
* Now normalize the curve
     smax=0.
     do 97 ka=1,51
     smax=max(smax,surcur(ka,ke,item))
  97 continue
     if(smax.ne.1.) then
     do 99 ka=1,51
     surcur(ka,ke,item)=surcur(ka,ke,item)/smax
  99 continue
     endif
*
 105 continue
```

```
* end of  "ke" loop over  16  environments
 115 continue
* end of "item"  loop over nritms  elements
*
    return
    end
*
    subroutine curexs(nritms,conseq,antand,antor,surcur,surexi,nbadc)
* Compute current exit survivabilities using data base status
*  25 August 1992
*
* Return from routine when the system's survivability membership curve
* surexi(..,..,nritms+1)  has been computed.
* (Redundant branches are not investigated)
*
* nritms  =  number of items (elements). "system" has the number nritms+1
* conseq(ka,kb) = ID-nrs of consequences of element "kb"
* antand(..,kb) = conjunctive antecedent ID-nrs of element "kb"
* antor(..,kb) = disjunctive antecedent ID-nrs of element "kb"
*  The ID-number 0 indicates source; ID-number 999 is the system;
*     ID-number 888 is dummy element (no element)
* surcur(51,16,kb) = survivability membership curves (51 nodes) for
*            16 environments of the element "kb"
*  The routine computes the following
* surexi(51,16,kb) = corresponding exit survivability membership curves
* nbadc = error indicator if system contains dead loops
*
    integer conseq(13,13),antand(13,13),antor(13,13)
    dimension surcur(51,16,13),surexi(51,16,13)
*
    integer scons(13,13),santan(13,13),santor(13,13)
    dimension itdone(13),orsum(51,16),ansum(51,16)
    dimension surlta(51),surltb(51),surgta(51),surgtb(51)
*
    nbadc=0
    do 15 ka=1,13
    itdone(ka)=0
    do 14 kb=1,13
    scons(kb,ka)=conseq(kb,ka)
    santan(kb,ka)=antand(kb,ka)
    santor(kb,ka)=antor(kb,ka)
  14 continue
  15 continue
*
* Find all elements that depend directly on the source
    ksor=0
    do 65 item=1,nritms+1
    kors=0
    do 21 ka=1,13
    if(santor(ka,item).eq.0) goto 35
    if(santor(ka,item).ne.888.and.santor(ka,item).ne.999)kors=kors+1
* count real "OR" dependencies
  21 continue
*
* Next take care of cases where source is (wrongly) given as "AND" antecedent
    ksor=888
```

```
      kands=0
      do 23 ka=1,13
      if(santan(ka,item).eq.999) santan(ka,item)=888
*  "system" (with ID 999) cannot be antecedent.  Replace it with dummy 888
      if(santan(ka,item).eq.0.and.ksor.eq.0) santan(ka,item)=888
*  Count "source" as antecedent only once
      if(santan(ka,item).ne.888) kands=kands+1
*  count valid "AND" antecedents
      if(santan(ka,item).eq.0) then
       ksor=0
       santan(ka,item)=888
      endif
   23 continue
*  Now have cancelled the source as an "AND" antecedent
      if(ksor.eq.0.and.kands.eq.1.and.kors.eq.0) goto 35
*  Go to 35 if the only "AND" antecedent is source and there are no "OR" antec.
      goto 65
*
   35 do 55 ke=1,16
      do 45 ks=1,51
      surexi(ks,ke,item)=surcur(ks,ke,item)
*  If element "item" is entered with an "OR" from source then other paths
*  can only transmit smaller consequences
   45 continue
   55 continue
      do 58 ka=1,13
      santan(ka,item)=888
      santor(ka,item)=888
*  If element depends on source then all other dependencies are dummies
   58 continue
      ksor=ksor+1
*  Count elements that depend on source
      itdone(item)=1
*  This is a list of "completed" elements with final exit survivabilities
*
      if(item.eq.nritms+1) return
*  Return if "system" outp ' has been computed (system is independent
*  of its elements, or has no elements)
   65 continue
*  End of loop over  "item"  from 1 to nritms+1
*  this took care of all elements with direct connection to source
      if(ksor.eq.0) then
      nbadc=1
      return
      endif
*  Error return: No source elements - graph consists of closed loops
*
      kloop=1
*  Counter of sweeps through the system
   77 continue
*  Next scan through all "completed" elements and establish consequences
*  Repeat until element with the ID-number nritms+1 is done
      do 152 kit=1,13
*
      if(itdone(kit).eq.0) goto 152
      item=kit
```

```
* found a done element "item"
    kcon=0
*  Next go through all consequence elements "icon"  of this "item"
  82 kcon=kcon+1
    icon=scons(kcon,item)
    if(icon.ne.888) goto 105
* Branch if "icon" is a real consequence "item"
    if(kcon.lt.13) goto 82
    goto 152
* Consequences of this "item" exhausted.  Branch to next "item"
* Check whether "icon" exit survivability can be computed or is already known
 105 if(icon.eq.999) icon=nritms+1
*  999 signifies the "system".  Its working ID-number is nritms+1
    if(itdone(icon).gt.0) then
    scons(kcon,item)=888
* This element "item" is already done.  Replace corresponding consequence by
* dummy number 888
    goto 82
    endif
*
* Now "icon" is a not-completed element.  See how it depends on others
* First combine all "OR" dependencies
    do 113 ka=1,16
    do 112 kb=1,51
    orsum(kb,ka)=0.
    ansum(kb,ka)=0.
 112 continue
    orsum(1,ka)=1.
    ansum(51,ka)=1.
 113 continue
*  Initial survivability memberships for the combinations
*
    kor=0
    do 122 ka=1,13
    if(santor(ka,icon).eq.888) goto 122
* Branch if OR antecedent is dummy (find OR antecedents for this icon)
     if(itdone(santor(ka,icon)).eq.0) goto 82
* go to the next not-completed consequence (loop 82) if source not completed
     do 118 ke=1,16
     call ltgt(orsum(1,ke),surlta,surgta)
     kors=santor(ka,icon)
     call ltgt(surexi(1,ke,kors),surltb,surgtb)
     do 117 kc=1,51
     orsum(kc,ke)=min(min(surgta(kc),surgtb(kc)),
   a            max(surlta(kc),surltb(kc)) )
*Combine "OR" dependencies
 117  continue
 118  continue
*  End of loop over 16 environments
*
     kor=kor+1
*  count "OR" dependencies
 122 continue
*  Next see if there are "AND" dependencies
    kand=0
    do 132 ka=1,13
```

```
    if(santan(ka,icon).eq.888) goto 132
      if(itdone(santan(ka,icon)).eq.0) goto 82
* Branch to loop 82 for next not-completed consequence because another
*  antecedent of this consequential element is not completed.
    kand=kand+1
     do 128 ke=1,16
     call ltgt(ansum(1,ke),surlta,surgta)
     kans=santan(ka,icon)
     call ltgt(surexi(1,ke,kans),surltb,surgtb)
     do 127 kc=1,51
     ansum(kc,ke)=min( min(surlta(kc),surltb(kc)),
   a            max(surgta(kc),surgtb(kc)) )
* Combine "AND" dependencies
 127  continue
 128  continue
 132 continue
*
* Combine the "ands" to the "ors" with "OR" to get a total input in orsum
    if(kand.gt.0) then
     do 138 ke=1,16
     call ltgt(orsum(1,ke),surlta,surgta)
     call ltgt(ansum(1,ke),surltb,surgtb)
*
     do 137 kb=1,51
     orsum(kb,ke)=min(min(surgta(kb),surgtb(kb)),
   a            max(surlta(kb),surltb(kb)) )
* Combine with "OR"
 137   continue
 138   continue
    endif
*
* Next combine the combined input with "AND" to "icon's" own survivability
* The result is the element's "icon" exit survivability 'surexi'
    do 143 ke=1,16
    call ltgt(orsum(1,ke),surlta,surgta)
    call ltgt(surcur(1,ke,icon),surltb,surgtb)
    do 142 kb=1,51
    surexi(kb,ke,icon)=min( min(surlta(kb),surltb(kb)),
   a               max(surgta(kb),surgtb(kb)) )
* Combine with "AND"
 142 continue
 143 continue
*
    itdone(icon)=1
* Indicate that exit survivability of element "icon" is computed
*
    if(icon.eq.nritms+1) return
* Return if the element "icon" was the system
    goto 82
* branch to find another not-completed consequence of "item"
*
 152 continue
*  end of loop "kit" over all completed elements
*
    kloop=kloop+1
    if(kloop.le.nritms+1) goto 77
```

```
*
      nbadc=kloop
* Error: System has not been reached in nritms+1 sweeps, i.e., "system"
*  cannot be reached from source.
      return
      end
*
      subroutine ltgt(amembr,surlta,surgta)
* Computes the possibilities of (less than a) and (greater than a).
* 26 August 1992
*
* amembr(51) = membership function of a
* surlta(51), surgta(51) = possibility distributions lt. a and gt. a
*
      dimension amembr(51),surlta(51),surgta(51)
*
      surgta(1)=amembr(1)
      surlta(51)=amembr(51)
      do 26 ka=2,51
      surgta(ka)=max(amembr(ka),surgta(ka-1))
      kb=52-ka
      surlta(kb)=max(amembr(kb),surlta(kb+1))
   26 continue
*
      return
      end
*
      subroutine nusurv(nritms,kasact,hrdact,surcur,hasurc)
* This computes modified element survivabilities due to hardening hrdact
*  27 August 1992
*
* nritms = number of elements in the system (excluding system itself)
* kasact(7) = ones indicate which hardening actions should be done
* hrdact(13,16,7) = hardenings [-5,5] for 13 elements in 16 environments.
*             and 7 hardening (modification) actions
* surcur(51,16,13) = present survivability membership curves: 51 entries,
*    16 environments, 13 elements
*
* The routine computes the following
* hasurc(51,16,13) = hardened (modified) survivability membership curves
*
      integer kasact(7),hrdact(13,16,7)
      dimension surcur(51,16,13),hasurc(51,16,13)
      dimension sxy(2,55)
*
      do 10 kit=1,nritms+1
      do 8  kenv=1,16
      do 6  ka=1,51
      hasurc(ka,kenv,kit)=surcur(ka,kenv,kit)
* This result is for "no hardening"
    6 continue
    8 continue
   10 continue
*
      do 104 kac=1,7
      if(kasact(kac).eq.0) goto 104
```

```
* Branch if the action "kac" is not active at time of this call
*
* Now apply the change hrdact to the survivability membership curves
    do 84 kit=1,nritms
    do 82 kenv=1,16
    if(hrdact(kit,kenv,kac).eq.0) goto 82
* Action "kac" for environment "kenv" and element "kit" is zero (no hardening)
*
*  Now harden the element "kit" for the environment "kenv"
*
    if(abs(hrdact(kit,kenv,kac)).eq.5) then
      do 12 ka=1,51
      hasurc(ka,kenv,kit)=0.
  12   continue
      if(hrdact(kit,kenv,kac).eq. 5) hasurc(51,kenv,kit)=1.
*  In this case have "absolutely" hardened the element
      if(hrdact(kit,kenv,kac).eq.-5) hasurc( 1,kenv,kit)=1.
*  In this case the modification is a softening that removes all protection
      goto 82
    endif
*
*  Next compute the left (raising) side of the hardened membership function
    kal=0
    do 24 kk=1,51
    if(hasurc(kk,kenv,kit).le.0.) goto 24
     if(kal.eq.0.and.hasurc(kk,kenv,kit).gt.0.005) then
       kal=1
       yalfa=0.005
*  Include a first  alpha-level  not higher than 0.005
       xalfa=float(kk-2)+yalfa/hasurc(kk,kenv,kit)
       hardl=float(hrdact(kit,kenv,kac))
       xalfa=xalfa + (hardl*0.2-(1.-yalfa)*0.10)*50.
** Left hand branch of hardening function!!
       sxy(1,kal)=xalfa
       sxy(2,kal)=yalfa
     endif
*
    kal=kal+1
    yalfa=hasurc(kk,kenv,kit)
    if(yalfa.ge.0.9999)yalfa=1.
    xalfa=float(kk-1)
* Now add the hardening at this alpha-level
    hardl=float(hrdact(kit,kenv,kac))
* hrdact = integer indicator of hardening category
    xalfa=xalfa + (hardl*0.2-(1.-yalfa)*0.10)*50.
** Left hand branch of membership function of hardening category!!
** Support width of hardening category is 0.20.
    sxy(1,kal)=xalfa
    sxy(2,kal)=yalfa
    kend=kk
    if(yalfa.ge.1.) goto 25
  24 continue
  25 kaltop=kal
* Increasing branch is now in  sxy(..,ka)  ka=1,kaltop
*
    if(sxy(1,1).ge.50.) then
```

```
      do 27 ka=1,50
      hasurc(ka,kenv,kit)=0.
  27   continue
      hasurc(51,kenv,kit)=1.
      goto 82
*  In this case the whole curve is shifted beyond x=50.
     endif
*
    if(sxy(1,kaltop).ge.50.) goto 45
*  Branch if decreasing part of the new curve is right of x-interval [0,50]
*
*  Next compute decreasing part of hardened membership function
    kxst=kend+1
    do 34 kx=kxst,51
    if(hasurc(kx,kenv,kit).le.0..and.hasurc(kx-1,kenv,kit).gt.0.)then
* Special treatment of last node
     if(hasurc(kx-1,kenv,kit).le.0.005) goto 45
*  Add one more node if last ordinate is larger than 0.005
     yalfa=0.005
     xalfa=float(kx)-yalfa/hasurc(kx-1,kenv,kit)
    else
     yalfa=hasurc(kx,kenv,kit)
     xalfa=float(kx-1)
    endif
  32 hardl=float(hrdact(kit,kenv,kac))
* hrdact = integer indicator of hardening level
    xalfa=xalfa + (hardl*0.2+(1.-yalfa)*0.10)*50.
** Right hand branch of membership function of hardening categories!!
    kal=kal+1
    sxy(1,kal)=xalfa
    sxy(2,kal)=yalfa
    if(yalfa.le.0.005) goto 45
  34 continue
* Hardened curve is now in sxy(..,ka), ka=1,kal
*
    if(sxy(1,kal).le.0.) then
      do 41 ka=2,51
      hasurc(ka,kenv,kit)=0.
  41   continue
      hasurc(1,kenv,kit)=1.
      goto 82
*  In this case the whole curve is shifted into negative x
     endif
*
*  Next interpolate for the 51 x-values using the function  sxf
  45  do 55 kx=1,51
    x=float(kx-1)
     if(x.lt.sxy(1,1).or.x.gt.sxy(1,kal)) then
      hasurc(kx,kenv,kit)=0.
      goto 55
     else
    do 48 ka=2,kal
      if(x.le.sxy(1,ka)) then
      f1=(sxy(1,ka)-x)/(sxy(1,ka)-sxy(1,ka-1))
      f2=(x-sxy(1,ka-1))/(sxy(1,ka)-sxy(1,ka-1))
      y=sxy(2,ka-1)*f1+sxy(2,ka)*f2
```

```
      hasurc(kx,kenv,kit)=y
      goto 55
      endif
 48  continue
     endif
*
 55  continue
*
     if(sxy(1,kaltop).le. 0.) hasurc( 1,kenv,kit)=1.
     if(sxy(1,kaltop).ge.50.) hasurc(51,kenv,kit)=1.
* Add crisp end-value if core is shifted outside x-interval [0,1]
*
* Next normalize the hardened (modified) curve
     hmax=0.
     do 66 ka=1,51
     hmax=max(hmax,hasurc(ka,kenv,kit))
  66 continue
     if(hmax.ne.1.) then
     do 69 ka=1,51
     hasurc(ka,kenv,kit)=hasurc(ka,kenv,kit)/hmax
  69 continue
     endif
*
  82 continue
* End of loop over "kenv" = environments
  84 continue
* End of loop over "kit" = elements
*
 104 continue
* end of loop over "kac" = actions

     return
     end
*
     subroutine stor(sysid,nritms,kase,kasact,cstact,hrdact,surexi,
    a inqui,inqsub,np1,mempl1,np2,mempl2,np3,mempl3)
* This interprets and stores the survivability of the system and
* associated costs in the files 'stor1-scap', 'stor2-scap' and 'stor3-scap'
* corresponding to the three inquiries "inqui"
* 28 August 1992
*
* sysid = system's AID (alphanumeric identification)
* nritms = number of items (elements) in the system
* kase = option ID-number
* kasact(7) = ones indicate which action was activated in this option
* cstact(7) = change of costs for each action [-3,3]
* hrdact(13,16,7) = hardening of 13 elements for 16 envrts by 7 actions
* surexi(51,16,13) = exit survivability curves (51 nodes) for
*       16 environments and 13 elements. System has ID-number nritms+1
* inqui(3) - Inquiry types: (1).ne.0 - need results for all environments
*                           (2).ne.0 - need summary for subset "inqsub"
*                           (3).ne.0 - need summary for all environments
* inqsub(16) - subset for inquiry-2 inqui(2).ne.0
* np1 - number of "inquiry 1" type curves to be plotted,
* mempl1(2,10) - the option number (mempl1(1,..)) and parameter
*           number (mempl1(2,..)) of the curve to be plotted
```

```
*  np2, np3 -  number of "inquiry 2" (or 3) type curves to be plotted
*  mempl2(10), mempl3(10)       -  the option numbers to be plotted
*
    character sysid*30
    integer kasact(7),cstact(7),hrdact(13,16,7),inqui(3),inqsub(16)
    dimension surexi(51,16,13),sursu2(51,1,1),sursu3(51,1,1)
   a ,mempl1(2,10),mempl2(10),mempl3(10)
*
    if(inqui(1).ne.0) call stor1(
   a        sysid,nritms,kase,kasact,cstact,hrdact,surexi)
     call stor2(inqui(2),
   a   sysid,nritms,kase,kasact,cstact,hrdact,surexi,inqsub,sursu2)
     call stor3(inqui(3),
   a       sysid,nritms,kase,kasact,cstact,hrdact,surexi,sursu3)
*
    if(npl+np2+np3.gt.0) call storpl(
   a        sysid,nritms,kase,surexi,sursu2,sursu3,
   b                   npl,mempl1,np2,mempl2,np3,mempl3)
*  Store for plotting the membership functions of the system
*   in the file 'stomem-scap'
*
    return
    end
*
    subroutine stor1(sysid,nritms,kase,kasact,cstact,hrdact,surexi)
*  Stores the answers to inquiry-1 (results for each environment parameter)
*  in 'stor1-scap' (unit 21) for human reading and in the file 'stan1-scap'
*  (unit 31) for later analysis.
*  14 September 1992
*
*  sysid = system's AID (alphanumeric identification)
*  nritms = number of items (elements) in the system
*  kase = option's ID-number
*  kasact(7) = ones indicate which action was activated in this option
*  cstact(7) = change of costs for each action  [-3,3]
*  hrdact(13,16,7) = hardening of 13 elements for 16 envrmts for 7 actions
*  surexi(51,16,13) = exit survivability curves (51 nodes) for
*        16 environments and 13 elements.  System has ID-number nritms+1
*
    character sysid*30
    integer kasact(7),cstact(7),hrdact(13,16,7)
    dimension surexi(51,16,13),coreor(16)
    integer musys(2,16),mcrior(16),kak(7),nritaf(13)
    character text*10,envlab(16)*24,surcat(8)*10,coscat(4)*8
    save musys,mcrior,coreor,envlab,surcat,coscat
    data envlab /'Over-pressure peak','Over-pressure impulse',
   a 'Dynamic-pressure peak','Dynamic-pressure impulse',
   b 'Under-pressure peak','Total thermal energy',
   c 'Maximum irradiance','Total dose, tissue','Total dose, silicon',
   d 'Total neutron dose','Neutron fluence','Total gamma dose',
   e 'Minimum threat yield','Maximum threat yield',
   f 'Ex-atmospheric EMP','Endo-atmospheric EMP'/
    data surcat /'  0.0  ','very poor','  poor  ',' moderate',
   a 'quite good','  good  ','very good','  1.0  '/
    data coscat /'  none  ',' small  ',' medium ',' large  '/
*
```

```
      if(kase.eq.0) then
      open(unit=21,file='stor1-scap')
      rewind(unit=21)
      goto 41
      else
      open(unit=21,file='stor1-scap',status='old')
 18    read(21,fmt=101,end=41,err=41) text
101 format(a1)
      goto 18
      endif
*
* Also open the unit 31 to store data for later analysis
 41    if(kase.eq.0) then
      open(unit=31,file='stan1-scap')
      rewind(unit=31)
* Next write preamble in the file
    write(31,501) sysid
501 format('Option Nr.  for system: 'a30)
    write(31,*) kase
    write(31,502)
502 format('Number of actions and action ID-numbers')
    write(31,*) 1,0
    write(31,504)
504 format('Costs  [-3,3]  for each action')
     write(31,*) 0
    write(31,506)
506 format('Number of affected elements and their ID-numbers')
    write(31,*) 1,0
    write(31,508)
508 format('Env. Nr., System surv.: low, high, core, mcrisp')
*
      goto 151
      else
      open(unit=31,file='stan1-scap',status='old')
 28    read(31,fmt=101,end=301,err=301) text
      goto 28
      endif
*
* Enter here for the unaltered results
151 write(21,152)
152 format(3x'SYSTEM SURVIVABILITIES FOR THE '
   a ,' 16 ENVIRONMENTS')
    write(21,155) sysid
155 format(/5x'NAME OF THE SYSTEM: 'a30)
*
    items=nritms+1
* This is the ID-number of the "system"
*
    write(21,157)
157 format(/'UNALTERED SURVIVABILITIES')
*
    do 172 kenv=1,16
    call interp(surexi,kenv,items,msurl,msurh,core,mcrisp)
* Interpret the survivability membership giving low and high survivabilities
* Code: 0 - crisp kill, 1-6 fuzzy survivab. categories, 7 - crisp survive
* Write this on unit 31
```

```
      write(31,*) kenv,msurl,msurh,core,mcrisp
*
      musys(1,kenv)=msurl
      musys(2,kenv)=msurh
      coreor(kenv)=core
      mcrior(kenv)=mcrisp
*
      write(21,161) kenv,envlab(kenv)
  161 format(/'ENVIRONMENT 'i2'.',2x,a24)
      if(msurl.eq.msurh) then
        if(mcrisp.eq.1.and.msurl.ne.0.and.msurl.ne.7) then
        write(21,162) core,surcat(msurl+1),msurl
  162   format(2x'Nuclear survivability:    '0pf6.2' or 'a10
     a ' ('i1')')
        else
        write(21,163) surcat(msurl+1),msurl
  163   format(2x'Nuclear survivability:    ',a10,' ('i1')')
        endif
      else
        if(mcrisp.eq.1) then
        write(21,164) core,surcat(msurl+1),msurl,surcat(msurh+1),msurh
  164   format(2x'Nuclear survivability:    '0pf6.2' or 'a10
     a  ' ('i1') to 'a10' ('i1')')
        else
        write(21,165) surcat(msurl+1),msurl,surcat(msurh+1),msurh
  165   format(2x'Nuclear survivability:    'a10' ('i1') to 'a10
     a  ' ('i1')')
        endif
      endif
*
  172 continue
*
      close(unit=21)
      close(unit=31)
      return
*
* Enter here to handle modifications
  301 write(21,305) kase
  305 format(/'MODIFICATION OPTION Nr.'i3'.')
      kn=0
      do 306 kk=1,7
      if(kasact(kk).ne.0) then
      kn=kn+1
      kak(kn)=kk
      endif
  306 continue
      write(21,310) (kak(j),j=1,kn)
  310 format('Modification actions: 'i1,6(', 'i1))
      write(21,315)
  315 format('Cost changes and affected elements:')
*
      do 340 jj=1,kn
      kk=kak(jj)
      kost=cstact(kk)
      if(kost.ge.0) then
      write(21,318) kk,coscat(kost+1),kost
```

```
 318 format(5x'Action Nr.'i2', cost increase 'a8' ('i1')')
     else
     write(21,319) kk,coscat(kost+1),kost
 319 format(5x'Action Nr.'i2', cost savings 'a8' ('i2')')
     endif
*
* Now find out numbers of affected elements by this action "kk"
     kit=0
     do 329 ka=1,13
     nokb=0
     do 328 kb=1,16
     if(nokb.eq.1) goto 328
     if(hrdact(ka,kb,kk).nc.0) then
     nokb=1
     kit=kit+1
     nritaf(kit)=ka
* Store in nritaf numbers of affected elements
     endif
 328 continue
 329 continue
     write(21,335) (nritaf(j),j=1,kit)
 335 format(5x,'Affected elements:'13(i3',',))
*
 340 continue
* Loop over "jj=1,kn" actions "kak(jj)" in this option "kcase"
*
* Next write preamble in the unit 31
     write(31,501) sysid
* 501 format('Option Nr. for system: 'a30)
     write(31,*) kase
     write(31,502)
* 502 format('Number of actions and action ID-numbers')
     write(31,*) kn,(kak(j),j=1,kn)
     write(31,504)
* 504 format('Costs [-3,3] for each action')
     write(31,*) (cstact(kak(j)),j=1,kn)
     write(31,506)
* 506 format('Number of affected elements and their ID-numbers')
     write(31,*) kit,(nritaf(j),j=1,kit)
     write(31,508)
* 508 format('Env. Nr., System surv.: low, high, core, mcrisp')
*
     do 372 kenv=1,16
     call interp(surexi,kenv,items,msurl,msurh,core,mcrisp)
* Interpret the survivability membership giving low and high survivabilities
* Code: 0 - crisp kill, 1-6 fuzzy survivab. categories, 7 - crisp survive
*
* Write this on unit 31
     write(31,*) kenv,msurl,msurh,core,mcrisp
*
     write(21,161) kenv,envlab(kenv)
*
     if(msurl.eq.msurh) then
       if(mcrisp.eq.1.and.msurl.ne.0.and.msurl.ne.7) then
       write(21,162) core,surcat(msurl+1),msurl
       else
```

```
      write(21,163) surcat(msurl+1),msurl
      endif
    else
      if(mcrisp.eq.1) then
      write(21,164) core,surcat(msurl+1),msurl,surcat(msurh+1),msurh
      else
      write(21,165) surcat(msurl+1),msurl,surcat(msurh+1),msurh
      endif
    endif
*
    jj=musys(1,kenv)
    jk=musys(2,kenv)
    jcp=mcrior(kenv)
    crs=coreor(kenv)
    if(jj.eq.jk) then
      if(jcp.eq.1.and.jj.ne.0.and.jj.ne.7) then
      write(21,362) crs, surcat(jj+1),jj
 362   format(2x'Original survivability was ',0pf6.2,' or 'a10
  a   ' ('i1')')
      else
      write(21,363) surcat(jj+1),jj
 363   format(2x'Original survivability was ',a10,' ('i1')')
      endif
    else
      if(jcp.eq.1.and.jj.ne.0.and.jj.ne.7) then
      write(21,364) crs, surcat(jj+1),jj,surcat(jk+1),jk
 364   format(2x'Original survivability was '0pf6.2' or 'a10
  a   ' ('i1') to 'a10' ('i1')')
      else
      write(21,365) surcat(jj+1),jj,surcat(jk+1),jk
 365   format(2x'Original survivability was 'a10' ('i1') to '
  a    a10' ('i1')')
      endif
    endif
*
 372 continue
* Loop over 16 "kenv" environments
*
    close(unit=21)
    close(unit=31)
    return
    end
*
    subroutine stor2(inq2,sysid,nritms,kase,kasact,cstact,hrdact,
   a surexi,inqsub,sursum)
* Stores the answers to inquiry-2 (summarized results for a subset
* of environments specified by "inqsub")
* The results are stored in 'stor2-scap' (unit 22) for human readers and
* stored in the file 'stan2-scap' (unit 32) for later analysis.
* 14 September 1992
*
* inq2 - if this is zero then only sursum is of interest (no storing)
* sysid = system's AID (alphanumeric identification)
* nritms = number of items (elements) in the system
* kase = option ID-number
* kasact(7) = ones indicate which action was activated in this option
```

```
* cstact(7) = change of costs for each action [-3,3]
* hrdact(13,16,7) = hardening of 13 elements for 16 envrmts by 7 actions
* surexi(51,16,13) = exit survivability curves (51 nodes) for
*       16 environments and 13 elements.  System has ID-number nritms+1
* inqsub(16) - subset of environments for which the result should be
*          summarized
* The routine computes
* sursum(51,1,1) - the combined membership function for the set
*
    character sysid*30
    integer kasact(7),cstact(7),hrdact(13,16,7),inqsub(16)
    dimension surexi(51,16,13)
   a ,sursum(51,1,1),surlta(51),surgta(51),surltb(51),surgtb(51)
    integer musub(2),kak(7),nritaf(13),inqnrs(16)
    character text*10,envlab(16)*24,surcat(8)*10,coscat(4)*8
    save musub,corsav,mcrsav,envlab,surcat,coscat
    data envlab /'Over-pressure peak','Over-pressure impulse',
   a 'Dynamic-pressure peak','Dynamic-pressure impulse',
   b 'Under-pressure peak','Total thermal energy',
   c 'Maximum irradiance','Total dose, tissue','Total dose, silicon',
   d 'Total neutron dose','Neutron fluence','Total gamma dose',
   e 'Minimum threat yield','Maximum threat yield',
   f 'Ex-atmospheric EMP','Endo-atmospheric EMP'/
    data surcat /'  0.0  ','very poor','  poor  ',' moderate',
   a 'quite good','  good  ','very good','  1.0  '/
    data coscat /'  none  ',' small  ',' medium ',' large  '/
*
    if(inq2.eq.0.and.kase.eq.0) goto 174
    if(inq2.eq.0.and.kase.gt.0) goto 341
* Branch if no storage:  only sursum is of interest.
* Open the output unit 22 for human readers
      if(kase.eq.0) then
      open(unit=22,file='stor2-scap')
      rewind(unit=22)
      goto 41
      else
      open(unit=22,file='stor2-scap',status='old')
 18    read(22,fmt=101,end=41,err=41) text
101 format(a1)
      goto 18
      endif
*
* Also open the unit 32 to store data for later analysis
 41    if(kase.eq.0) then
      open(unit=32,file='stan2-scap')
      rewind(unit=32)
      goto 151
      else
      open(unit=32,file='stan2-scap',status='old')
 28    read(32,fmt=101,end=301,err=301) text
      goto 28
      endif
*
* Enter here for the unaltered results
151    write(22,153)
153 format(3x'SYSTEM SURVIVABILITY W/R TO A',
```

```
   a ' SUBSET OF ENVIRONMENTS'/)
      write(22,155) sysid
 155 format(5x'NAME OF THE SYSTEM: 'a30,/)
    write(22,156)
 156 format('The subset consists of the following parameters:'/)
*
    kinq=0
    do 172 kenv=1,16
    if(inqsub(kenv).eq.0) goto 172
    kinq=kinq+1
    inqnrs(kinq)=kenv
    if(inq2.gt.0) write(22,161) kenv,envlab(kenv)
 161 format('Environment 'i2'.',2x,a24)
 172 continue
*
 174 do 175 ka=1,50
    sursum(ka,1,1)=0.
 175 continue
    sursum(51,1,1)=1.
    nrsys=nritms+1
* This is the ID-number of the "system"
*
    do 192 kenv=1,16
* Next combine the exit survivabilities "surexi" of the appropriate envir.
* with logical "AND" into sursum
    if(inqsub(kenv).eq.0) goto 192
    call ltgt(sursum,surlta,surgta)
    call ltgt(surexi(1,kenv,nrsys),surltb,surgtb)
    do 187 kc=1,51
    sursum(kc,1,1)=min( min(surlta(kc),surltb(kc)),
   a          max(surgta(kc),surgtb(kc)) )
 187 continue
 192 continue
*
    if(inq2.eq.0) return
* Return if only sursum is of interest: no storing
*
    call interp(sursum,1,1,msurl,msurh,core,mcrisp)
* Interpret the survivability membership giving low and high survivabilities
* Code: 0 - crisp kill, 1-6 fuzzy survivab. categories, 7 - crisp survive
*
    musub(1)=msurl
    musub(2)=msurh
    corsav=core
    mcrsav=mcrisp
*
    if(msurl.eq.msurh) then
      if(mcrisp.eq.1.and.msurl.ne.0.and.msurl.ne.7) then
      write(22,194) core,surcat(msurl+1),msurl
 194   format(/2x'Original survivability:   ',0pf6.2,' or 'a10
   a  ' ('i1')')
      else
      write(22,195) surcat(msurl+1),msurl
 195   format(/2x'Original survivability:   'a10' ('i1')')
      endif
    else
```

```
      if(mcrisp.eq.1) then
      write(22,196)core,surcat(msurl+1),msurl,surcat(msurh+1),msurh
  196   format(/2x'Original survivability:   '0pf6.2' or 'a10
     a  ' ('i2') to 'a10' ('i2')')
      else
      write(22,197) surcat(msurl+1),msurl,surcat(msurh+1),msurh
  197   format(/2x'Original survivability:   'a10' ('i1') to 'a10
     a ' ('i1')')
      endif
    endif
*
    close(unit=22)
    goto 401
*
* Enter here to handle modifications
  301 write(22,305) kase
  305 format(/'MODIFICATION OPTION Nr.'i3'.')
    kn=0
    do 306 kk=1,7
    if(kasact(kk).ne.0) then
    kn=kn+1
    kak(kn)=kk
    endif
  306 continue
    write(22,310) (kak(j),j=1,kn)
  310 format('Modification actions: 'i1,6(', 'i1))
    write(22,315)
  315 format('Cost changes and affected elements:')
*
    do 340 jj=1,kn
    kk=kak(jj)
    kost=cstact(kk)
    if(kost.ge.0) then
    write(22,318) kk,coscat(kost+1),kost
  318 format(5x'Action Nr.'i2', cost increase 'a8' ('i1')')
    else
    write(22,319) kk,coscat(kost+1),kost
  319 format(5x'Action Nr.'i2', cost savings 'a8' ('i2')')
    endif
*
* Now find out numbers of affected elements by this action "kk"
    kit=0
    do 329 ka=1,13
    nokb=0
    do 328 kb=1,16
    if(nokb.eq.1) goto 328
    if(hrdact(ka,kb,kk).ne.0) then
    nokb=1
    kit=kit+1
    nritaf(kit)=ka
    endif
* Store in nritaf numbers of affected elements
  328 continue
  329 continue
    write(22,335) (nritaf(j),j=1,kit)
  335 format(5x,'Affected elements:'13(i3','))
```

```
*
 340 continue
*
 341 do 352 ka=1,51
    sursum(ka,1,1)=0.
 352 continue
    sursum(51,1,1)=1.
    nrsys=nritms+1
* This is the ID-number of the "system"
*
    do 357 kenv=1,16
* Next combine the exit survivabilities "surexi" of the appropriate envir.
* with logical "AND" into sursum
    if(inqsub(kenv).eq.0) goto 357
    call ltgt(sursum,surlta,surgta)
    call ltgt(surexi(1,kenv,nrsys),surltb,surgtb)
    do 355 kc=1,51
    sursum(kc,1,1)=min( min(surlta(kc),surltb(kc)),
   a          max(surgta(kc),surgtb(kc)) )
 355 continue
 357 continue
*
    if(inq2.eq.0) return
* Return if only sursum is of interest
*
    call interp(sursum,1,1,msurl,msurh,core,mcrisp)
* Interpret the survivability membership giving low and high survivabilities
* Code: 0 - crisp kill, 1-6 fuzzy survivab. categories, 7 - crisp survive
* mcrisp=1 indicates that the membership sursum is crisp
*
    if(msurl.eq.msurh) then
      if(mcrisp.eq.1.and.msurl.ne.0.and.msurl.ne.7) then
      write(22,364) core,surcat(msurl+1),msurl
 364   format(/2x'Nuclear survivability:    ',0pf6.2,' or 'a10
  a   ' ('i1')')
      else
      write(22,365) surcat(msurl+1),msurl
 365   format(/2x'Nuclear survivability:    'a10' ('i1')')
      endif
    else
      if(mcrisp.eq.1) then
      write(22,366) core,surcat(msurl+1),msurl,surcat(msurh+1),msurh
 366   format(/2x'Nuclear survivability:    '0pf6.2' or 'a10
  a   ' ('i2') to 'a10' ('i2')')
      else
      write(22,367) surcat(msurl+1),msurl,surcat(msurh+1),msurh
 367   format(/2x'Nuclear survivability:    'a10' ('i1') to 'a10
  a ' ('i1')')
      endif
    endif
*
    jj=musub(1)
    jk=musub(2)
    if(jj.eq.jk) then
      if(mcrsav.eq.1.and.jj.ne.0.and.jj.ne.7) then
      write(22,382) corsav,surcat(jj+1),jj
```

```
 382   format(2x'Original survivability was '0pf6.2' or 'a10
  a  ' ('i1')')
      else
      write(22,383) surcat(jj+1),jj
 383   format(2x'Original survivability was ',a10,' ('i1')')
      endif
    else
      if(mcrsav.eq.1) then
      write(22,384) corsav, surcat(jj+1),jj,surcat(jk+1),jk
 384   format(2x'Original survivability was '0pf6.2' or 'a10
  a  ' ('i1') to 'a10' ('i1')')
      else
      write(22,385) surcat(jj+1),jj,surcat(jk+1),jk
 385   format(2x'Original survivability was 'a10' ('i1') to 'a10
  a  ' ('i1')')
      endif
    endif
*
    close(unit=22)
*
* Enter here and store the results in unit 32 for later analysis
 401 if(kase.eq.0) then
     write(32,495)
 495   format('Combined result for: Nr of envs., env. ID-numbers')
     write(32,*) kinq, (inqnrs(j),j=1,kinq)
    endif
    write(32,501) sysid
 501 format('Option Nr.  for system: 'a30)
    write(32,*) kase
    write(32,502)
 502 format('Number of actions and action ID-numbers')
    if(kase.eq.0) write(32,*) 1,0
    if(kase.ne.0) write(32,*) kn,(kak(j),j=1,kn)
    write(32,504)
 504 format('Costs  [-3,3]  for each action')
    if(kase.eq.0) write(32,*) 0
    if(kase.ne.0) write(32,*) (cstact(kak(j)),j=1,kn)
    write(32,506)
 506 format('Number of affected elements and their ID-numbers')
    if(kase.eq.0) write(32,*) 1,0
    if(kase.ne.0) write(32,*) kit,(nritaf(j),j=1,kit)
    write(32,508)
 508 format('System survivability: low, high, core, mcrisp')
    write(32,*) msurl,msurh,core,mcrisp

    close(unit=32)
    return
    end
*
    subroutine stor3(inq3,sysid,nritms,kase,kasact,cstact,hrdact,
   a surexi,sursum)
* Stores the answers to inquiry-3 (summarized results for all
* environments)
* The results are stored in 'stor3-scap' (unit 23) for human readers and
* in 'stan3-scap' (unit 33) for later analysis.
* 14 September 1992
```

```
*
* inq3 = if this is 0 then no storage: only sursum is of interest
* sysid = system's AID (alphanumeric identification)
* nritms = number of items (elements) in the system
* kase = option's ID-number
* kasact(7) = ones indicate which action was activated in this option
* cstact(7) = change of costs for each action [-3,3]
* hrdact(13,16,7) = hardening of 13 elements for 16 envrmts by 7 actions
* surexi(51,16,13) = exit survivability curves (51 nodes) for
*       16 environments and 13 elements. System has ID-number nritms+1
* The routine computes
* sursum(51,1,1) = The system exit survivability membership function
*
    character sysid*30
    integer kasact(7),cstact(7),hrdact(13,16,7)
    dimension surexi(51,16,13)
   a ,sursum(51,1,1),surita(51),surgta(51),surltb(51),surgtb(51)
    integer muall(2),kak(7),nritaf(13)
    character text*10,envlab(16)*24,surcat(8)*10,coscat(4)*8
    save muall,corsav,mcrsav,envlab,surcat,coscat
    data envlab /'Over-pressure peak','Over-pressure impulse',
   a 'Dynamic-pressure peak','Dynamic-pressure impulse',
   b 'Under-pressure peak','Total thermal energy',
   c 'Maximum irradiance','Total dose, tissue','Total dose, silicon',
   d 'Total neutron dose','Neutron fluence','Total gamma dose',
   e 'Minimum threat yield','Maximum threat yield',
   f 'Ex-atmospheric EMP','Endo-atmospheric EMP'/
    data surcat /'  0.0  ','very poor','   poor',' moderate',
   a 'quite good','   good','very good','  1.0  '/
    data coscat /' none ',' small ',' medium ',' large '/
*
    if(inq3.eq.0.and.kase.eq.0) goto 171
    if(inq3.eq.0.and.kase.gt.0) goto 350
* Skip all sorting operations if only sursum is of interest
*
      if(kase.eq.0) then
      open(unit=23,file='stor3-scap')
      rewind(unit=23)
      goto 41
      else
      open(unit=23,file='stor3-scap',status='old')
 18    read(23,fmt=101,end=41,err=41) text
101 format(a1)
      goto 18
      endif
*
* Also open the unit 33 to store data for for later analysis
 41    if(kase.eq.0) then
      open(unit=33,file='stan3-scap')
      rewind(unit=33)
      goto 151
      else
      open(unit=33,file='stan3-scap',status='old')
 28    read(33,fmt=101,end=301,err=301) text
      goto 28
      endif
```

```
*
* Enter here for the unaltered results
 151   write(23,159)
 159 format(2x'COMBINED SYSTEM SURVIVABILITY W/R TO ALL '
  a ,'ENVIRONMENTS')
     write(23,155) sysid
 155 format(/3x'NAME OF THE SYSTEM: 'a30,/)
*
 171 do 175 ka=1,50
   sursum(ka,1,1)=0.
 175 continue
   sursum(51,1,1)=1.
   nrsys=nritms+1
* This is the ID-number of the "system"
*
   do 192 kenv=1,16
* Next combine the exit survivabilities "surexi" of all environments
* with logical "AND" into sursum
   call ltgt(sursum,surlta,surgta)
   call ltgt(surexi(1,kenv,nrsys),surltb,surgtb)
   do 187 kc=1,51
   sursum(kc,1,1)=min( min(surlta(kc),surltb(kc)),
  a          max(surgta(kc),surgtb(kc)) )
 187 continue
 192 continue
*
   if(inq3.eq.0) return
* Return if only sursum is of interest
*
   call interp(sursum,1,1,msurl,msurh,core,mcrisp)
* Interpret the survivability membership giving low and high survivabilities
* Code: 0 - crisp kill, 1-6 fuzzy survivab. categories, 7 - crisp survive
* mcrisp=1 indicates that the membership sursum is crisp
*
   muall(1)=msurl
   muall(2)=msurh
   corsav=core
   mcrsav=mcrisp
*
   if(msurl.eq.msurh) then
     if(mcrisp.eq.1.and.msurl.ne.0.and.msurl.ne.7) then
     write(23,194) core,surcat(msurl+1),msurl
 194   format(5x'Original survivability: ',0pf6.2,' or 'a10
  a  ' ('i1')')
     else
     write(23,195) surcat(msurl+1),msurl
 195   format(5x'Original survivability: ',a10,' ('i1')')
     endif
   else
     if(mcrisp.eq.1) then
     write(23,196) core,surcat(msurl+1),msurl,surcat(msurh+1),msurl
 196   format(5x'Original survivability: '0pf6.2' or 'a10
  a  ' ('i1') to 'a10' ('i1')')
     else
     write(23,197) surcat(msurl+1),msurl,surcat(msurh+1),msurh
 197   format(5x'Original survivability: 'a10' ('i1') to 'a10
```

```
   a  ' ('i1')')
     endif
    endif
*
    close(unit=23)
    goto 401
*
* Enter here to handle modifications
 301 write(23,305) kase
 305 format(/'MODIFICATION OPTION Nr.'i3'.')
    kn=0
    do 309 kk=1,7
     if(kasact(kk).ne.0) then
     kn=kn+1
     kak(kn)=kk
     endif
 309 continue
    write(23,310) (kak(j),j=1,kn)
 310 format('Modification actions: 'i1,6(', 'i1))
    write(23,312)
 312 format('Cost changes and affected elements:')
*
    do 340 jj=1,kn
    kk=kak(jj)
    kost=cstact(kk)
    if(kost.ge.0) then
    write(23,315) kk,coscat(kost+1),kost
 315 format(5x'Action Nr.'i2',  cost increase 'a8' ('i1')')
    else
    write(23,316) kk,coscat(kost+1),kost
 316 format(5x'Action Nr.'i2',  cost savings 'a8' ('i2')')
    endif
*
* Now find out numbers of affected elements by this action "kk"
    kit=0
    do 319 ka=1,13
    nokb=0
    do 318 kb=1,16
    if(nokb.eq.1) goto 318
    if(hrdact(ka,kb,kk).ne.0) then
    nokb=1
    kit=kit+1
    nritaf(kit)=ka
    endif
* Store in nritaf numbers of affected elements
 318 continue
 319 continue
    write(23,325) (nritaf(j),j=1,kit)
 325 format(5x,'Affected elements:'13(i3',',))
 340 continue
*
 350 do 352 ka=1,51
    sursum(ka,1,1)=0.
 352 continue
    sursum(51,1,1)=1.
    nrsys=nritms+1
```

```
* This is the ID-number of the "system"
*
    do 357 kenv=1,16
* Next combine the exit survivabilities "surexi" of all environments
* with logical "AND" into sursum
    call ltgt(sursum,surlta,surgta)
    call ltgt(surexi(1,kenv,nrsys),surltb,surgtb)
    do 355 kc=1,51
    sursum(kc,1,1)=min( min(surlta(kc),surltb(kc)),
   a          max(surgta(kc),surgtb(kc)) )
 355 continue
 357 continue
*
    if(inq3.eq.0) return
* Return if only  sursum  is of interest
*
    call interp(sursum,1,1,msurl,msurh,core,mcrisp)
* Interpret the survivability membership giving low and high survivabilities
* Code: 0 - crisp kill, 1-6 fuzzy survivab. categories, 7 - crisp survive
* mcrisp=1 indicates that the membership  sursum  is crisp
*
    if(msurl.eq.msurh) then
      if(mcrisp.eq.1.and.msurl.ne.0.and.msurl.ne.7) then
      write(23,364) core,surcat(msurl+1),msurl
 364   format(/5x'Nuclear survivability:    ',0pf6.2,' or 'a10
  a  ' ('i1')')
      else
      write(23,365) surcat(msurl+1),msurl
 365   format(/5x'Nuclear survivability:    'a10' ('i1')')
      endif
    else
      if(mcrisp.eq.1) then
      write(23,366) core,surcat(msurl+1),msurl,surcat(msurh+1),msurh
 366   format(/5x'Nuclear survivability:    '0pf6.2' or 'a10
  a  ' ('i2') to 'a10' ('i2')')
      else
      write(23,367) surcat(msurl+1),msurl,surcat(msurh+1),msurh
 367   format(/5x'Nuclear survivability:    'a10' ('i1') to 'a10
  a ' ('i1')')
      endif
    endif
*
    jj=muall(1)
    jk=muall(2)
    if(jj.eq.jk) then
      if(mcrsav.eq.1.and.jj.ne.0.and.jj.ne.7) then
      write(23,382) corsav, surcat(jj+1),jj
 382   format(5x'Original survivability was ',0pf6.2,' or 'a10
  a   ' ('i1')')
      else
      write(23,383) surcat(jj+1),jj
 383   format(5x'Original survivability was ',a10,' ('i1')')
      endif
    else
      if(mcrsav.eq.1) then
      write(23,384) corsav, surcat(jj+1),jj,surcat(jk+1),jk
```

```
 384    format(5x'Original survivability was '0pf6.2' or 'a10
  a    ' ('i1') to 'a10' ('i1')')
       else
       write(23,385) surcat(jj+1),jj,surcat(jk+1),jk
 385    format(5x'Original survivability was 'a10' ('i1') to 'a10
  a    ' ('i1')')
       endif
     endif
*
     close(unit=23)
*
* Enter here and store the results in unit 33 for later analysis
 401 write(33,501) sysid
 501 format('Option Nr.  for system: 'a30)
     write(33,*) kase
     write(33,502)
 502 format('Number of actions and action ID-numbers')
     if(kase.eq.0) write(33,*) 1,0
     if(kase.ne.0) write(33,*) kn,(kak(j),j=1,kn)
     write(33,504)
 504 format('Costs  [-3,3]  for each action')
     if(kase.eq.0) write(33,*) 0
     if(kase.ne.0) write(33,*) (cstact(kak(j)),j=1,kn)
     write(33,506)
 506 format('Number of affected elements and their ID-numbers')
     if(kase.eq.0) write(33,*) 1,0
     if(kase.ne.0) write(33,*) kit,(nritaf(j),j=1,kit)
     write(33,508)
 508 format('System survivability: low, high, core, mcrisp')
     write(33,*) msurl,msurh,core,mcrisp
*
     close(unit=33)
     return
     end
*
     subroutine interp(surexi,kenv,item,msurl,msurh,core,mcrisp)
* Interpret the survivability membership by computing a low and a high
* survivability bound of surexi for environment "kenv" and element "item"
*  28 August 1992
*
* surexi(51,16,13) = exit survivability curves (51 nodes) for
*    16 environments and 13 elements.  System is element Nr. is nritms+1
* kenv =  environment ID-number
* item = item (element) ID-number
*
* The routine will compute the following from the survivability membership
*  curve
* msurl = low (fuzzy) linguistic category of survivability
* msurh = high (fuzzy) linguistic category of survivability
*      According to the following code:
*      0 - crisp kill, 1-6 fuzzy survivab. categories, 7 - crisp survivab.
* core = leftmost x-coordinate of the core
*       ("core" is only used in output if surexi is crisp)
* mcrisp = fuzziness indicator: 0 if surexi is fuzzy, 1 if surexi is crisp
*
     dimension surexi(51,16,13)
```

```
      dimension catint(6)
*
      y(x,xz)=max(0.,1.-abs(x-xz)/(0.16*50.))
** Membership of standard survivability categories!! (For core = xz)
** Assume a support width of 0.32.  Unit length is 50 (51 nodes)
*
* First find the core
      surmax=0.
      do 18 ka=1,51
      if(surexi(ka,kenv,item).gt.surmax) then
        kcore=ka
        core=float(ka-1)/50.
* leftmost x-coordinate of core
      endif
      surmax=max(surmax,surexi(ka,kenv,item))
   18 continue
*
* Next find out whether this is a crisp number
      mcrisp=1
      do 19 ka=1,51
      if(ka.eq.kcore) goto 19
      if(surexi(ka,kenv,item).gt.0..and.
     a  surexi(ka,kenv,item).le.surmax) then
        mcrisp=0
        goto 21
      endif
   19 continue
*
* Next take care of the two extremes (crisp kill and crisp survive)
   21 if(mcrisp.eq.1.and.kcore.eq.1) then
        msurl=0
        msurh=0
        return
      endif
      if(mcrisp.eq.1.and.kcore.eq.51) then
        msurl=7
        msurh=7
        return
      endif
*
* Next compute the threshold Lambda for assigning survivability categories
      thresh=0.
      xz1=0.4*50.
      xz2=xz1+0.1*50.
* Cores for standard survivabilities "3" and "3.5"!!
      do 34 ka=15,30
      x=float(ka)
      thresh=thresh+min(y(x,xz1),y(x,xz2))
   34 continue
* This threshold is the intersection area between two standard membership
* functions apart exactly 1/2 distance between the six standard categories
*
* Next determine the intersections of "surexi" with the six standard
*  survivability category membership functions
      cinmax=0.
      do 48 kat=1,6
```

```
      xzc=float(kat-1)*0.2*50.
** Cores of standard survivability categories!!
      catint(kat)=0.
      do 42 ka=1,51
      x=float(ka)
      catint(kat)=catint(kat)+min(y(x,xzc),surexi(ka,kenv,item))
   42 continue
       if(catint(kat).gt.cinmax) then
       katmax=kat
       cinmax=catint(kat)
       endif
   48 continue
*  Maximum intersection is with "katmax" standard, intersection value is cinmax
*
*  Next determine msurl = lowest above-threshold intersection with standard cat.
      do 52 ka=1,katmax
      msurl=ka
      if(catint(ka).gt.thresh) goto 54
   52 continue
* and msurh = highest above-threshold intersection with standard survivability
   54 msurh=katmax
      do 56 ka=katmax,6
      if(catint(ka).gt.thresh) msurh=ka
   56 continue
*
      return
      end
*
      subroutine inprnt(envir,sysid,nract,cstact,hrdact,
     a  nritms,itemid,surcur,surexi,inqui,inqsub)
*  Writes in "stin-scap" neatly printable summary of input describing the
*  state of the system and actions that are to be investigated.
*  16 September 1992
*
      dimension envir(3,16)
      integer cstact(7),hrdact(13,16,7),inqui(3),inqsub(16)
*
      character sysid*30,itemid(13)*30,envlab(16)*24,surcat(8)*10,
     a  coscat(4)*8,surlev(3)*10
*
      dimension surcur(51,16,13),surexi(51,16,13)
      integer nritaf(13)
      save envlab,surcat,coscat
*
*  envir(3,16) - environment: (l,c,r) for 16 environment parameters
*  sysid - system AID  (alphanumeric identification)
*  nract - number of actions to be investigated
*  cstact(7) - costs of 7 proposed actions
*  hrdact(13,16,7)  - hardening categories of the actions (13 items,
*               16 environments, 7 actions)
*  inqui(3)  -  specification of questions asked (3 types of inquiry)
*  inqsub(16) -  subset (list) of environment parameters that is to be analyzed
*               if  inqui(2).ne.0
*  nritms  -  number of items (elements) in the system
*  itemid(13) - item AIDs
* surcur(51,16,13) - survivability membership functions for 16 env. & 13 elem.
```

```
* surexi(51,16,13) - exit surv. memberships for 16 envir. and 13 items (elem)
* surexi(51,16,nritms+1) - exit survivability of the system for 16 envir.
*
    data envlab /'Over-pressure peak','Over-pressure impulse',
   a 'Dynamic-pressure peak','Dynamic-pressure impulse',
   b 'Under-pressure peak','Total thermal energy',
   c 'Maximum irradiance','Total dose, tissue','Total dose, silicon',
   d 'Total neutron dose','Neutron fluence','Total gamma dose',
   e 'Minimum threat yield','Maximum threat yield',
   f 'Ex-atmospheric EMP','Endo-atmospheric EMP'/
    data surcat /'  0.0  ','very poor','  poor  ',' moderate',
   a 'quite good','  good  ','very good','  1.0  '/
    data coscat /'  none  ',' small  ',' medium ',' large  '/
*
    open(unit=12,file='stin-scap')
    rewind(unit=12)
*
* First print the present state of the system
*
    npage=0
    do 79 ka=1,nritms+1,3
    npage=npage+1
    if(npage.eq.1) write(12,20) npage
 20 format('    INPUT SUMMARY'48x'Page 'i1)
    if(npage.gt.1) write(12,21) npage
 21 format('   INPUT SUMMARY'48x'Page 'i1)
   write(12,22) sysid,nritms,nritms+1
 22 format(/'    NAME OF THE SYSTEM: 'a30/
  a ' Number of elements in the system: 'i2/
  a ' Element Nr.'i2' is the system.')
   write(12,25)
 25 format(/48x'Nuclear Survivabilities')
   ilow=ka
   ihig=min(ka+2,nritms+1)
   nr=1+ihig-ilow
   if(nr.eq.1) write(12,27) (j,j=ilow,ihig)
   if(nr.eq.2) write(12,28) (j,j=ilow,ihig)
   if(nr.eq.3) write(12,29) (j,j=ilow,ihig)
 27 format(8x'Parameter'12x'Value (l,c,r)'4x'Element'i2,2x)
 28 format(8x'Parameter'12x'Value (l,c,r)'4x,2('Element'i2,3x))
 29 format(8x'Parameter'12x'Value (l,c,r)'4x,3('Element'i2,3x))
*
   do 72 ken=1,16
   kp=0
   do 56 kit=ilow,ihig
   kp=kp+1
    if(kit.ne.nritms+1) then
    call interp(surcur,ken,kit,msurl,msurh,core,mcrisp)
    else
    call interp(surexi,ken,kit,msurl,msurh,core,mcrisp)
    endif
      if(mcrisp.eq.1) then
      write(surlev(kp),36) core
 36    format(0pf6.2,4x)
      else
      surlev(kp)=surcat(msurl+1)
```

```
* write name of lower bound of survivability membership
      endif
 56 continue
*
   write(12,59) ken,envlab(ken),(envir(j,ken),j=1,3),
  a (surlev(j),j=1,kp)
 59 format(/i2'.',1x,a24,3(1x,0pf4.0),3(2x,a10))
*
 72 continue
*
 79 continue
*
   npage=npage+1
   write(12,21) npage
* Next print the proposed actions
   write(12,85)
 85 format(/'  MODIFICATIONS')
   do 153 kact=1,nract
   ncost=cstact(kact)
     if(ncost.lt.0) then
     write(12,94) kact,coscat(-ncost+1)
 94   format(/'ACTION Nr. 'i1'.'3x'Cost savings 'a8)
     else
     write(12,95) kact,coscat(ncost+1)
 95   format(/'ACTION Nr. 'i1'.'3x'Cost increase: 'a8)
     endif
   write(12,100)
100 format(/5x'Element'20x'Environment parameters')
   write(12,101)(j,j=1,16)
101 format(25x'Survivability changes [-5,+5]'/13x,16(1x,i2))
*
* Now find out numbers of affected elements by this action "kact"
   kit=0
   do 129 ka=1,13
   nokb=0
   do 128 kb=1,16
   if(nokb.eq.1) goto 128
   if(hrdact(ka,kb,kact).ne.0) then
   nokb=1
   kit=kit+1
   nritaf(kit)=ka
* Store in nritaf numbers of affected elements
   endif
128 continue
129 continue
   do 136 ka=1,kit
   kitem=nritaf(ka)
   write(12,132) kitem,itemid(kitem),(hrdact(kitem,j,kact),j=1,16)
132 format(i1'. 'a10,16(1x,i2))
136 continue
*
153 continue
* End of loop over actions "kact"
*
   close(unit=12)
   return
```

```
      end
*
      subroutine anal1
* Reads results from 'stan1-scap' (unit 31), finds the best options and
* writes them into the file 'stopt1-scap' (unit 41)
*
      dimension nract(127),nrsact(7,127),ncosts(7,127),nritm(127),
     a nrsitm(13,127),lhsurv(2,16,127),cores(16,127),mcrisp(16,127)
     b ,nopt(127),nopcst(127),nopt5(5)
     c ,msyslo(16),msyshi(16),corsys(16),mcrsys(16)
      character sysid*30,text*1,surcat(8)*10,coscat(4)*8,envlab(16)*24
      save envlab,surcat,coscat
      data envlab /'Over-pressure peak','Over-pressure impulse',
     a 'Dynamic-pressure peak','Dynamic-pressure impulse',
     b 'Under-pressure peak','Total thermal energy',
     c 'Maximum irradiance','Total dose, tissue','Total dose, silicon',
     d 'Total neutron dose','Neutron fluence','Total gamma dose',
     e 'Minimum threat yield','Maximum threat yield',
     f 'Ex-atmospheric EMP','Endo-atmospheric EMP'/
      data surcat /'  0.0  ','very poor','  poor  ',' moderate',
     a 'quite good','  good  ','very good','  1.0  '/
      data coscat /'  none  ',' small  ',' medium ',' large  '/
*
        open(unit=31,file='stan1-scap')
        rewind(unit=31)
*
* First read and store the unaltered result, "kase=0"
      read(31,501) sysid
  501 format(22x,a30)
      do 55 ka=1,8
      read(31,52) text
   52  format(a1)
   55 continue
      do 58 kenv=1,16
      read(31,*) ken,msyslo(ken),msyshi(ken),corsys(ken),mcrsys(ken)
* Low and high survivabilities, core survivability, crisp number indicator
   58 continue
*
* Next read the data for all "kases" options
*
      kases=0
   61 read(31,52,end=71,err=71) text
      kases=kases+1
      read(31,*) kase
* Option number
      read(31,52) text
      read(31,*) nract(kase),(nrsact(j,kase),j=1,nract(kase))
* Number of activated actions, and action numbers
      read(31,52) text
      read(31,*) (ncosts(j,kase),j=1,nract(kase))
* Costs of the activated actions in this option (kase)
      read(31,52) text
      read(31,*) nritm(kase),(nrsitm(j,kase),j=1,nritm(kase))
* Number of elements, and id-numbers of elements affected by this option
      read(31,52) text
      do 63 kenv=1,16
```

```
      read(31,*) ken,lhsurv(1,ken,kase),lhsurv(2,ken,kase),
     a cores(ken,kase),mcrisp(ken,kase)
*  Low and high survivability, core of survivab., crisp surv. indicator
    63 continue
      goto 61

   71 close(unit=31)
*
*  Now have read all data "kases" (options).
      open(unit=41,file='stoptl-scap')
      rewind(unit=41)
*  Open result file
*
*  Write header on result file
      write(41,72)
    72 format(' '10x'LIST OF BEST OPTIONS W/R TO EACH'
     a ,' ENVIRONMENT')
      write (41,73) sysid
    73 format(/'NAME OF THE SYSTEM: 'a30)
*
    74 do 551 kenv=1,16
*
*  Arrange output according to 16 environment parameters
      write(41,76) kenv,envlab(kenv)
    76 format(/'ENVIRONMENT PARAMETER: 'i2'. 'a24)
*
      write(41,78)
    78 format('The unaltered system survivability is')
*
      if(msyslo(kenv).eq.msyshi(kenv)) then
        if(mcrsys(kenv).eq.1.and.msyslo(kenv).ne.0.
     a                  and.msyslo(kenv).ne.7) then
        write(41,162) corsys(kenv),surcat(msyslo(kenv)+1),msyslo(kenv)
   162   format(6x,0pf6.2' or 'a10' ('i1')')
        else
        write(41,163) surcat(msyslo(kenv)+1),msyslo(kenv)
   163   format(6x,a10,' ('i1')')
        endif
      else
        if(mcrsys(kenv).eq.1) then
        write(41,164) corsys(kenv),surcat(msyslo(kenv)+1),msyslo(kenv),
     a              surcat(msyshi(kenv)+1),msyshi(kenv)
   164   format(6x,0pf6.2' or 'a10' ('i1') to 'a10' ('i1')')
        else
        write(41,165) surcat(msyslo(kenv)+1),msyslo(kenv),
     a          surcat(msyshi(kenv)+1),msyshi(kenv)
   165   format(6x,a10' ('i1') to 'a10 ' ('i1')')
        endif
      endif
*
*  Next find the option with largest lower bound of survivability,
*                                lhsurv(1,kenv,kase)
      maxlo=msyslo(kenv)
      minlo=msyslo(kenv)
      do 201 ka=1,kases
      maxlo=max(maxlo,lhsurv(1,kenv,ka))
```

```
      minlo=min(minlo,lhsurv(1,kenv,ka))
  201 continue
*
      if(maxlo.le.msyslo(kenv).and.minlo.ge.msyslo(kenv)) then
      write(41,202)
  202 format('None of the suggested options changes the lower bound'
     a ,' of survivability')
      goto 551
      endif
*
      write(41,199)
  199 format(/'Options that produce the highest',
     a ' lower bounds of survivability are'/)
*
* Next find all those options that have this lower bound of survivability.
* List them and select the 5 cheapest options.
      kopt=0
      nclow=7
      do 221 ka=1,kases
      if(lhsurv(1,kenv,ka).lt.maxlo) goto 221
* Now store the numbers of options with comparable survivabilities in nopt
      kopt=kopt+1
      nopt(kopt)=ka
* Next compute and store the associated number of cost sources in nopcst
      nc=0
      do 205 kb=1,nract(ka)
      if(ncosts(kb,ka).gt.0) nc=nc+1
      if(ncosts(kb,ka).lt.0) nc=nc-1
* Note that only the number of costs and savings counts, not the size of costs
  205 continue
      nopcst(kopt)=nc
      nclow=min(nclow,nc)
* nclow is the lowest cost level of the acceptable actions
  221 continue
*
      kop5=kopt
* Next find the 5 cheapest options and store their numbers in nopt5(5)
      kop5=0
  225 nclow1=7
* nclow1 will be the next (higher) cost level to be considered
      do 241 ka=1,kopt
      if(nopcst(ka).le.nclow) then
      kop5=kop5+1
      nopt5(kop5)=nopt(ka)
      if(kop5.ge.min(5,kopt)) goto 251
* Take the first 5 that meet the cheapness level
      nopcst(ka)=8
* This prevents the choice of the same action as cheap the next time around
      goto 241
      endif
      nclow1=min(nclow1,nopcst(ka))
  241 continue
      nclow=nclow1
      goto 225
*
  251 write(41,252)
```

```
 252 format('Option'2x'Survivability'15x,'Action Nr.'5x'Cost changes')
 256 do 451 ka=1,kop5
   nop=nopt5(ka)
*
   if(lhsurv(1,kenv,nop).eq.lhsurv(2,kenv,nop)) then
     if(mcrisp(kenv,nop).eq.1.and.lhsurv(1,kenv,nop).ne.0.and.
  a             lhsurv(1,kenv,nop).ne.7) then
     write(41,262) nop,cores(kenv,nop),surcat(lhsurv(1,kenv,nop)+1),
  a                            lhsurv(1,kenv,nop)
 262  format(/i3,2x,0pf6.2' or 'a10' ('i1')')
     else
     write(41,263) nop,surcat(lhsurv(1,kenv,nop)+1),
  a                            lhsurv(1,kenv,nop)
 263  format(/i3,2x,a10,' ('i1')')
     endif
   else
     if(mcrisp(kenv,nop).eq.1) then
     write(41,264) nop,cores(kenv,nop),surcat(lhsurv(1,kenv,nop)+1)
  a      ,lhsurv(1,kenv,nop),surcat(lhsurv(2,kenv,nop)+1),
  b                            lhsurv(2,kenv,nop)
 264  format(/i3,2x,0pf6.2' or 'a10' ('i1') to 'a10' ('i1')')
     else
     write(41,265) nop,surcat(lhsurv(1,kenv,nop)+1),
  a  lhsurv(1,kenv,nop),surcat(lhsurv(2,kenv,nop)+1),
  b                            lhsurv(2,kenv,nop)
  a
 265  format(/i3,2x,a10' ('i1') to 'a10 ' ('i1')')
     endif
   endif
*
   do 395 kb=1,nract(nop)
   nco=ncosts(kb,nop)
   if(nco.gt.0) write(41,388) nrsact(kb,nop),coscat(nco+1)
   if(nco.eq.0) write(41,390) nrsact(kb,nop),coscat(nco+1)
   if(nco.lt.0) write(41,392) nrsact(kb,nop),coscat(nco+1)
 388 format(40x,i2,5x,a8' increase')
 390 format(40x,i2,5x,a8)
 392 format(40x,i2,5x,a8' savings')
 395 continue
*
 451 continue
* End of loop over best options (statement 256)
*
 551 continue
* End of loop  over 16 environment parameters "kenv" (statement 74)
*
   close(unit=41)
   return
   end
*
   subroutine anal2
* Reads results from 'stan2-scap' (unit 32), finds the best options and
* writes them into the file 'stopt2-scap' (unit 42)
*
   dimension nract(127),nrsact(7,127),ncosts(7,127),nritm(127),
  a nrsitm(13,127),lhsurv(2,127),cores(127),mcrisp(127),nrsenv(16)
```

```
     b ,nopt(127),nopcst(127),nopt5(5)
      character sysid*30,text*1,surcat(8)*10,coscat(4)*8,envlab(16)*24
      save envlab,surcat,coscat
      data envlab /'Over-pressure peak','Over-pressure impulse',
     a 'Dynamic-pressure peak','Dynamic-pressure impulse',
     b 'Under-pressure peak','Total thermal energy',
     c 'Maximum irradiance','Total dose, tissue','Total dose, silicon',
     d 'Total neutron dose','Neutron fluence','Total gamma dose',
     e 'Minimum threat yield','Maximum threat yield',
     f 'Ex-atmospheric EMP','Endo-atmospheric EMP'/
      data surcat /'  0.0  ','very poor','  poor  ',' moderate',
     a 'quite good','  good  ','very good','  1.0  '/
      data coscat /'  none  ',' small  ',' medium ',' large  '/
*
        open(unit=32,file='stan2-scap')
        rewind(unit=32)
*
* First read and store the unaltered result, "kase=0"
      read(32,52) text
      read(32,*) nrenv,(nrsenv(j),j=1,nrenv)
* Numbers of the environment over which the combined result is sought
      read(32,501) sysid
  501 format(22x,a30)
      do 55 ka=1,8
      read(32,52) text
   52 format(a1)
   55 continue
      read(32,*) msyslo,msyshi,corsys,mcrsys
* Low and high survivabilities, core survivability, crisp number indicator
*
* Next read the data for all "kases" options
*
      kases=0
   61 read(32,52,end=71,err=71) text
      kases=kases+1
      read(32,*) kase
* Option number
      read(32,52) text
      read(32,*) nract(kase),(nrsact(j,kase),j=1,nract(kase))
* Number of activated actions, and action numbers
      read(32,52) text
      read(32,*) (ncosts(j,kase),j=1,nract(kase))
* Costs of the activated actions in this option (kase)
      read(32,52) text
      read(32,*) nritm(kase),(nrsitm(j,kase),j=1,nritm(kase))
* Number of elements, and id-numbers of elements affected in this option
      read(32,52) text
      read(32,*) lhsurv(1,kase),lhsurv(2,kase),cores(kase),mcrisp(kase)
* Low and high survivability, core of survivab., crisp survivab. indicator
      goto 61

   71 close(unit=32)
*
* Now have read all data "kases" (options).
      open(unit=42,file='stopt2-scap')
      rewind(unit=42)
```

```
* Open result file
*
* Write header on result file
    write(42,72)
   72 format(3x'LIST OF BEST OPTIONS W/R TO A SUBSET OF'
    a ,' ENVIRONMENTS')
    write(42,73) sysid
   73 format(/5x'NAME OF THE SYSTEM: 'a30)
    write(42,74)
   74 format(/'The result is combined for the following environments:')
    do 77 ka=1,nrenv
    jj=nrsenv(ka)
    write(42,76) jj,envlab(jj)
   76 format(10x,i2,'. 'a24)
   77 continue
    write(42,78)
   78 format(/'The unaltered system survivability is')
*
    if(msyslo.eq.msyshi) then
      if(mcrsys.eq.1.and.msyslo.ne.0.and.msyslo.ne.7) then
      write(42,162) corsys,surcat(msyslo+1),msyslo
  162   format(5x,0pf6.2' or 'a10' ('i1')')
      else
      write(42,163) surcat(msyslo+1),msyslo
  163   format(5x,a10,' ('i1')')
      endif
    else
      if(mcrsys.eq.1) then
      write(42,164) corsys,surcat(msyslo+1),msyslo,
     a              surcat(msyshi+1),msyshi
  164   format(5x,0pf6.2' or 'a10' ('i1') to 'a10' ('i1')')
      else
      write(42,165) surcat(msyslo+1),msyslo,surcat(msyshi+1),msyshi
  165   format(5x,a10' ('i1') to 'a10 ' ('i1')')
      endif
    endif
*
* Next find the option with largest lower bound of surviv, lhsurv(1,kase)
    lowmax=msyslo
    lowmin=msyslo
    do 201 ka=1,kases
    lowmax=max(lowmax,lhsurv(1,ka))
    lowmin=min(lowmin,lhsurv(1,ka))
  201 continue
*
    if(lowmax.le.msyslo.and.lowmin.eq.msyslo) then
    write(42,202)
  202 format('None of the suggested actions changes the lower bound'
    a ,' of the survivability')
    goto 455
    endif
*
    write(42,203)
  203 format(/'For the following options the lower bound of'
    a ,' survivability is largest'/)
*
```

```
* Next find all those options that have this lower bound of survivability.
* List them and select the 5 cheapest options.
    kopt=0
    nclow=7
    do 221 ka=1,kases
    if(lhsurv(1,ka).lt.lowmax) goto 221
* Now store the numbers of options with comparable survivabilities in nopt
    kopt=kopt+1
    nopt(kopt)=ka
* Next compute and store the associated number of cost sources in nopcst
    nc=0
    do 205 kb=1,nract(ka)
    if(ncosts(kb,ka).gt.0) nc=nc+1
    if(ncosts(kb,ka).lt.0) nc=nc-1
* Note that only the number of costs and savings counts, not the size of costs
 205 continue
    nopcst(kopt)=nc
* This is the cost of this action
    nclow=min(nclow,nc)
* nclow contains the lowest cost of acceptable actions
 221 continue
*
    kop5=kopt
* Next find the 5 cheapest options and store their numbers in nopt5(5)
    kop5=0
 225 nclow1=7
* nclow1 will be the next (higher) cost level to be considered
    do 241 ka=1,kopt
    if(nopcst(ka).le.nclow) then
    kop5=kop5+1
    nopt5(kop5)=nopt(ka)
    if(kop5.ge.min(5,kopt)) goto 261
* Take the first 5 that meet the cheapness level
    nopcst(ka)=8
* This avoids that the same action will be found the next time around
    goto 241
    endif
    nclow1=min(nclow1,nopcst(ka))
* nclow1 contains the next lowest level of costs
 241 continue
    nclow=nclow1
    goto 225
*
 261 write(42,271)
 271 format('Option'2x'Survivability'15x,'Action Nr.'5x'Cost changes')
*
    do 451 ka=1,kop5
    nop=nopt5(ka)
*
    if(lhsurv(1,nop).eq.lhsurv(2,nop)) then
      if(mcrisp(nop).eq.1.and.lhsurv(1,nop).ne.0.and.
   a             lhsurv(1,nop).ne.7) then
      write(42,262) nop,cores(nop),surcat(lhsurv(1,nop)+1),
   a                          lhsurv(1,nop)
 262   format(/i3,2x,0pf6.2' or 'a10' ('i1')')
      else
```

```
      write(42,263) nop,surcat(lhsurv(1,nop)+1),lhsurv(1,nop)
  263   format(/i3,2x,a10,' ('i1')')
      endif
    else
      if(mcrisp(nop).eq.1) then
      write(42,264) nop,cores(nop),surcat(lhsurv(1,nop)+1),
   a       lhsurv(1,nop),surcat(lhsurv(2,nop)+1),lhsurv(2,nop)
  264   format(/i3,2x,0pf6.2' or 'a10' ('i1') to 'a10' ('i1')')
      else
      write(42,265) nop,surcat(lhsurv(1,nop)+1),lhsurv(1,nop),
   a           surcat(lhsurv(2,nop)+1),lhsurv(2,nop)
  265   format(/i3,2x,a10' ('i1') to 'a10 ' ('i1')')
      endif
    endif
*
    do 395 kb=1,nract(nop)
    nco=ncosts(kb,nop)
    if(nco.gt.0) write(42,388) nrsact(kb,nop),coscat(nco+1)
    if(nco.eq.0) write(42,390) nrsact(kb,nop),coscat(nco+1)
    if(nco.lt.0) write(42,392) nrsact(kb,nop),coscat(nco+1)
  388 format(40x,i2,5x,a8' increase')
  390 format(40x,i2,5x,a8)
  392 format(40x,i2,5x,a8' savings')
  395 continue
*
  451 continue
*
  455 close(unit=42)
    return
    end
*
    subroutine anal3
* Reads results from 'stan3-scap' (unit 33), finds the best options and
* writes them into the file 'stopt3-scap' (unit 43).
*
    dimension nract(127),nrsact(7,127),ncosts(7,127),nritm(127),
   a nrsitm(13,127),lhsurv(2,127),cores(127),mcrisp(127)
   b ,nopt(127),nopcst(127),nopt5(10)
    character sysid*30,text*1,surcat(8)*10,coscat(4)*8
    save surcat,coscat
    data surcat /'  0.0  ','very poor','  poor ',' moderate',
   a 'quite good','  good ','very good','  1.0  '/
    data coscat /' none ',' small ',' medium ',' large '/
*
      open(unit=33,file='stan3-scap')
      rewind(unit=33)
*
* First read and store the unaltered result, "kase=0"
    read(33,501) sysid
  501 format(22x,a30)
    do 55 ka=1,8
    read(33,52) text
  52  format(a1)
   55 continue
    read(33,*) msyslo,msyshi,corsys,mcrsys
* Low and high survivabilities, core survivability, crisp number indicator
```

```
*
* Next read the data for all "kases" options
*
    kases=0
 61 read(33,52,end=71,err=71) text
    kases=kases+1
    read(33,*) kase
* Option number
    read(33,52) text
    read(33,*) nract(kase),(nrsact(j,kase),j=1,nract(kase))
* Number of activated actions, and action numbers
    read(33,52) text
    read(33,*) (ncosts(j,kase),j=1,nract(kase))
* Costs of the activated actions in this option (kase)
    read(33,52) text
    read(33,*) nritm(kase),(nrsitm(j,kase),j=1,nritm(kase))
* Number of elements, and id-numbers of elements affected in this option (kase)
    read(33,52) text
    read(33,*) lhsurv(1,kase),lhsurv(2,kase),cores(kase),mcrisp(kase)
* Low and high survivability, core of survivab., crisp survivab. indicator
    goto 61

 71 close(unit=33)
*
* Now have read all data (the number of options is "kases" ).
    open(unit=43,file='stopt3-scap')
    rewind(unit=43)
* Open result file
*
* Write header on result file
    write(43,72)
 72 format(3x'LIST OF BEST OPTIONS FOR OVERALL SYSTEM'
   a ,' SURVIVABILITY')
    write(43,73) sysid
 73 format(/5x'NAME OF THE SYSTEM: 'a30)
    write(43,78)
 78 format(/'The unaltered system survivability is')
*
    if(msyslo.eq.msyshi) then
      if(mcrsys.eq.1.and.msyslo.ne.0.and.msyslo.ne.7) then
      write(43,162) corsys,surcat(msyslo+1),msyslo
162    format(5x,0pf6.2' or 'a10' ('i1')')
      else
      write(43,163) surcat(msyslo+1),msyslo
163    format(5x,a10,' ('i1')')
      endif
    else
      if(mcrsys.eq.1) then
      write(43,164) corsys,surcat(msyslo+1),msyslo,
   a                surcat(msyshi+1),msyshi
164    format(5x,0pf6.2' or 'a10' ('i1') to 'a10' ('i1')')
      else
      write(43,165) surcat(msyslo+1),msyslo,surcat(msyshi+1),msyshi
165    format(5x,a10' ('i1') to 'a10 ' ('i1')')
      endif
    endif
```

```
*
* Next find the options with largest lower bound of surviv, lhsurv(1,kase)
    lowmax=msyslo
    lowmin=msyslo
    do 201 ka=1,kases
    lowmax=max(lowmax,lhsurv(1,ka))
    lowmin=min(lowmin,lhsurv(1,ka))
 201 continue
*
    if(lowmax.le.msyslo.and.lowmin.eq.msyslo) then
    write(43,202)
 202 format('None of the proposed options changes the lower bound of'
   a ,' the survivability')
    goto 455
    endif
*
    write(43,199)
 199 format(/'For the following options the lower bound of',
   a ' survivability is largest'/)
*
* Next find all those options that have this lower bound of survivability.
* List them and select the 10 cheapest options.
    kopt=0
    nclow=7
    do 221 ka=1,kases
    if(lhsurv(1,ka).lt.lowmax) goto 221
* Now store the numbers of options with comparable survivabilities in nopt
    kopt=kopt+1
    nopt(kopt)=ka
* Next compute and store the associated number of cost sources in nopcst
    nc=0
    do 205 kb=1,nract(ka)
    if(ncosts(kb,ka).gt.0) nc=nc+1
    if(ncosts(kb,ka).lt.0) nc=nc-1
* Note that only the number of costs and savings counts, not the size of costs
 205 continue
    nopcst(kopt)=nc
    nclow=min(nclow,nc)
 221 continue
* nclow contains the lowest cost level of acceptable actions
*
    kop5=kopt
* Next find the 5 cheapest options and store their numbers in nopt5(5)
    kop5=0
 225 nclow1=7
* nclow1 will be the next (higher) cost level to be considered
    do 241 ka=1,kopt
    if(nopcst(ka).le.nclow) then
    kop5=kop5+1
    nopt5(kop5)=nopt(ka)
    if(kop5.ge.min(10,kopt)) goto 261
* Take the first 10 that meet the cheapness level
    nopcst(ka)=8
* This prevents the selection of this action the next time around
    goto 241
    endif
```

```
      nclow1=min(nclow1,nopcst(ka))
  241 continue
      nclow=nclow1
      goto 225
*
  261 write(43,271)
  271 format('Option'2x'Survivability'15x,'Action Nr.'5x'Cost changes')
*
      do 451 ka=1,kop5
      nop=nopt5(ka)
*
      if(lhsurv(1,nop).eq.lhsurv(2,nop)) then
        if(mcrisp(nop).eq.1.and.lhsurv(1,nop).ne.0.and.
     a                  lhsurv(1,nop).ne.7) then
        write(43,362) nop,cores(nop),surcat(lhsurv(1,nop)+1),
     a                           lhsurv(1,nop)
  362   format(/i3,2x,0pf6.2' or 'a10' ('i1')')
        else
        write(43,363) nop,surcat(lhsurv(1,nop)+1),lhsurv(1,nop)
  363   format(/i3,2x,a10,' ('i1')')
        endif
      else
        if(mcrisp(nop).eq.1) then
        write(43,364) nop,cores(nop),surcat(lhsurv(1,nop)+1),
     a       lhsurv(1,nop),surcat(lhsurv(2,nop)+1),lhsurv(2,nop)
  364   format(/i3,2x,0pf6.2' or 'a10' ('i1') to 'a10' ('i1')')
        else
        write(43,365) nop,surcat(lhsurv(1,nop)+1),lhsurv(1,nop),
     a            surcat(lhsurv(2,nop)+1),lhsurv(2,nop)
  365   format(/i3,2x,a10' ('i1') to 'a10 ' ('i1')')
        endif
      endif
*
      do 395 kb=1,nract(nop)
      nco=ncosts(kb,nop)
      if(nco.gt.0) write(43,388) nrsact(kb,nop),coscat(nco+1)
      if(nco.eq.0) write(43,390) nrsact(kb,nop),coscat(nco+1)
      if(nco.lt.0) write(43,392) nrsact(kb,nop),coscat(nco+1)
  388 format(40x,i2,5x,a8' increase')
  390 format(40x,i2,5x,a8)
  392 format(40x,i2,5x,a8' savings')
  395 continue
*
  451 continue
*
  455 close(unit=43)
      return
      end
*
      subroutine storpl(sysid,nritms,kase,surexi,sursu2,sursu3,
     a np1,mempl1,np2,mempl2,np3,mempl3)
*  Stores membership functions of the system in the file 'stomem-scap'
*  (unit 7) for plotting
*   1 October 1992
*
*  sysid - system AID  (alphanumeric identification)
```

```
*  nritms  - number of elements (items) in the system.  Nr. "nritms+1"
*        is the system and will be plotted
*  kase -  number of option
*  surexi(51,16,13) - survivability membership functions for 16 environments
*               and 13 items
*  sursu2(51,1,1)  -  combined subset survivability
*  sursu3(51,1,1)  -  combined system survivability
*  np1 - number of "inquiry 1" type curves to be plotted,
*  mempl1(2,10) - the option number (mempl1(1,..)) and parameter
*            number (mempl1(2,..))  of the curve to be plotted
*  np2, np3 -  number of "inquiry 2" (or inq. 3)-type curves to be plotted
*  mempl2(10), mempl3(10)   -  corresponding option numbers to be plotted
*
     character sysid*30,text*1
     dimension surexi(51,16,13),sursu2(51,1,1),sursu3(51,1,1)
     integer mempl1(2,10),mempl2(10),mempl3(10)
*
     if(kase.eq.0) then
       open(unit=7,file='stomem-scap')
       rewind(unit=7)
     else
       open(unit=7,file='stomem-scap',status='old')
  18    read(7,fmt=19,end=41,err=41) text
  19 format(a1)
       goto 18
     endif
*
     nsys=nritms+1
     if(kase.ne.0) goto 41
*  Store system AID
     write(7,20) sysid
  20 format(a30)
*  Store plotting information
     write(7,*) np1,mempl1
     write(7,*) np2,mempl2
     write(7,*) np3,mempl3
*
*  Store initial membership curves for all environments
     do 23 ka=1,16
     call interp(surexi,ka,nsys,msurl,msurh,core,mcrisp)
     write(7,*) 0,0,ka,msurl,msurh,core,mcrisp
     write(7,*) (surexi(j,ka,nsys),j=1,51)
  23 continue
*
*  Store initial membership curve for subset of environments (inquiry 2)
     call interp(sursu2,1,1,msurl,msurh,core,mcrisp)
     write(7,*) 0,0,22,msurl,msurh,core,mcrisp
     write(7,*) (sursu2(j,1,1),j=1,51)
*  Note:  If there was no inquiry 2  then  sursu2  is a crisp unity.
*  Store membership curve combined for all parameters
     call interp(sursu3,1,1,msurl,msurh,core,mcrisp)
     write(7,*) 0,0,23,msurl,msurh,core,mcrisp
     write(7,*) (sursu3(j,1,1),j=1,51)
*  sursu3 is the system survivability
*
     close(unit=7)
```

```
      return
*
   41 if(np1.eq.0) goto 51
      do 45 ka=1,np1
      if(kase.eq.mempl1(1,ka)) then
      do 43 kb=1,16
      if(kb.eq.mempl1(2,ka)) then
      call interp(surexi,kb,nsys,msurl,msurh,core,mcrisp)
      write(7,*) 1,kase,kb,msurl,msurh,core,mcrisp
      write(7,*)(surexi(j,kb,nsys),j=1,51)
*  Store the membership curves requested by inquiry 1
      endif
   43 continue
      endif
   45 continue
*
   51 if(np2.eq.0) goto 61
      do 55 ka=1,np2
      if(kase.eq.mempl2(ka)) then
      call interp(sursu2,1,1,msurl,msurh,core,mcrisp)
      write(7,*) 2,kase,0,msurl,msurh,core,mcrisp
      write(7,*) (sursu2(j,1,1),j=1,51)
*  Store the membership curves requested by inquiry 2
      endif
   55 continue
*
   61 if(np3.eq.0) goto 71
      do 65 ka=1,np3
      if(kase.eq.mempl3(ka)) then
      call interp(sursu3,1,1,msurl,msurh,core,mcrisp)
      write(7,*) 3,kase,0,msurl,msurh,core,mcrisp
      write(7,*) sursu3
*  Store the membership curves requested by inquiry 3
      endif
   65 continue
*
   71 close(unit=7)
      return
      end
```

INTENTIONALLY LEFT BLANK

Appendix D.

DATA BANK ENTRY FOR THE "SYSTEM 3a"

INTENTIONALLY LEFT BLANK

```
System AID (alphanumeric ID) in format(a30) :
System 3a
Number of items (elements) in this system:
3
Item ID-No., Item AID; format(i2,a30)
 1  3a - 1
Consequences: number, Item ID-No's. (888 is none, 999 is system)
1  3
Conjunctive antecedents: number, Item ID-No's (0 is source, 888 is none)
1  0
Disjunctive antecedents: number, Item ID-No's  (0 is source, 888 is none)
1  0
Survivabilities: Parameter ID-No., t1,t2,t3,  b1,b2,b3
1   20.  20.  20.  60.  60.  60.
2   22.  23.  24.  61.  62.  63.
3   24.  26.  28.  62.  64.  66.
4   26.  29.  32.  63.  66.  69.
5   28.  32.  36.  64.  68.  72.
6   30.  35.  40.  65.  70.  75.
7   32.  38.  44.  66.  72.  78.
8   34.  41.  48.  67.  74.  81.
9   36.  44.  52.  68.  76.  84.
10  38.  47.  56.  69.  78.  87.
11  40.  50.  60.  70.  80.  90.
12  42.  53.  64.  71.  82.  93.
13  44.  56.  68.  72.  84.  96.
14  60.  80.  90.  80.  100.  110.
15  60.  80.  90.  80.  100.  110.
16  60.  80.  90.  80.  100.  110.
Item ID-No., Item AID; format(i2,a30)
 2  3a - 2
2-Consequences: number, Item ID-No's. (888 is none, 999 is system)
1  3
Conjunctive antecedents: number, Item ID-No's  (0 is source, 888 is none)
1  0
Disjunctive antecedents: number, Item ID-No's  (0 is source, 888 is none)
1  0
Survivabilities: Parameter ID-No., t1,t2,t3,  b1,b2,b3
1   20.  20.  20.  60.  60.  60.
2   22.  23.  24.  61.  62.  63.
3   24.  26.  28.  62.  64.  66.
4   26.  29.  32.  63.  66.  69.
5   28.  32.  36.  64.  68.  72.
6   30.  35.  40.  65.  70.  75.
7   32.  38.  44.  66.  72.  78.
8   34.  41.  48.  67.  74.  81.
9   36.  44.  52.  68.  76.  84.
10  38.  47.  56.  69.  78.  87.
11  40.  50.  60.  70.  80.  90.
12  42.  53.  64.  71.  82.  93.
13  44.  56.  68.  72.  84.  96.
14  46.  59.  72.  73.  86.  99.
15  48.  62.  76.  74.  88.  102.
16  50.  65.  80.  75.  80.  105.
```

```
Item ID-No., Item AID; format(i2,a30)
 3 3a - 3
Consequences: number, Item ID-No's. (888 is none, 999 is system)
1 999
Conjunctive antecedents: number, Item ID-No's (0 is source, 888 is none)
1 888
Disjunctive antecedents: number, Item ID-No's (0 is source, 888 is none)
2 1 2
Survivabilities: Parameter ID-No., t1,t2,t3, b1,b2,b3
1  20. 20. 20. 60. 60. 60.
2  22. 23. 24. 61. 62. 63.
3  24. 26. 28. 62. 64. 66.
4  26. 29. 32. 63. 66. 69.
5  28. 32. 36. 64. 68. 72.
6  30. 35. 40. 65. 70. 75.
7  32. 38. 44. 66. 72. 78.
8  34. 41. 48. 67. 74. 81.
9  36. 44. 52. 68. 76. 84.
10 38. 47. 56. 69. 78. 87.
11 40. 50. 60. 70. 80. 90.
12 42. 53. 64. 71. 82. 93.
13 44. 56. 68. 72. 84. 96.
14 66. 79. 92. 93. 106. 119.
15 68. 82. 96. 94. 108. 122.
16 70. 85. 100. 95. 100. 125.
Item ID-No., Item AID; format(i2,a30)
 4 3a - System
Consequences: number, Item ID-No's. (888 is none, 999 is system)
1 888
Conjunctive antecedents: number, Item ID-No's (0 is source, 888 is none)
1 3
Disjunctive antecedents: number, Item ID-No's (0 is source, 888 is none)
1 888
Survivabilities: Parameter ID-No., t1,t2,t3, b1,b2,b3
1  100. 100. 100. 100. 100. 100.
2  100. 100. 100. 100. 100. 100.
3  100. 100. 100. 100. 100. 100.
4  100. 100. 100. 100. 100. 100.
5  100. 100. 100. 100. 100. 100.
6  100. 100. 100. 100. 100. 100.
7  100. 100. 100. 100. 100. 100.
8  100. 100. 100. 100. 100. 100.
9  100. 100. 100. 100. 100. 100.
10 100. 100. 100. 100. 100. 100.
12 100. 100. 100. 100. 100. 100.
11 100. 100. 100. 100. 100. 100.
13 100. 100. 100. 100. 100. 100.
14 100. 100. 100. 100. 100. 100.
15 100. 100. 100. 100. 100. 100.
16 100. 100. 100. 100. 100. 100.
```

Appendix E.

CASE INPUT FILE FOR THE "SYSTEM 3a"

INTENTIONALLY LEFT BLANK

```
Sample input August 92
Environment: Parameter ID-No., value (low bound, center, high bound)
1  50.  50.  50.
2  50.  50.  50.
3  50.  50.  50.
4  50.  50.  50.
5  50.  50.  50.
6  50.  50.  50.
7  50.  50.  50.
8  50.  50.  50.
9  51.  52.  55.
10 52.  54.  60.
11 53.  56.  65.
12 54.  58.  70.
13 55.  60.  75.
14 56.  62.  80.
15 57.  64.  85.
16 58.  66.  90.
System AID in format(a30):
System 3a
Number of actions to be analyzed
3
Action ID-No., number of items changed, change of costs [-3,3]
1  3  2
Hardening: Item ID-No., 16 hardness changes [-5,5]
1 2 0 0 0 0 0 0 0 0 0 0 0 2 0 0 1
2 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 1
3 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Action ID-No., number of items changed, change of costs [-3,3]
2  2  1
Hardening: Item ID-No., 16 hardness changes [-5,5]
1 2 0 0 0 0 0 0 0 0 0 0 0 2 0 0 2
3 2 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0
Action ID-No., number of items changed, change of costs [-3,3]
3  2  3
Hardening: Item ID-nr., 16 hardness changes [-5,5]
2 2 2 2 2 0 0 0 0 0 0 4 4 3 3 3 3
3 2 2 3 3 0 0 0 0 0 0 4 4 3 0 0 0
Inquiries: Each [0,1], Subset [0, number of parameters in set], All [0,1]
           1          2                1
Subset (if inquiry(2) is positive). One dummy number if inquiry(2)=0.
12  13
Number of membership curves that should be plotted
5
Inquiry type, option, parameter (needed if type=1; dummy if type=2 or 3)
1          1     1
1          6     16
2          4     0
3          1     0
3          7     0
```

INTENTIONALLY LEFT BLANK

Appendix F.

SUMMARY OF INPUT FOR THE "SYSTEM 3a"

INTENTIONALLY LEFT BLANK

INPUT SUMMARY

NAME OF THE SYSTEM: System 3a
Number of elements in the system: 3
Element Nr. 4 is the system.

Parameter	Value (l,c,r)			Nuclear Survivabilities Element 1	Element 2	Element 3
1. Over-pressure peak	50.	50.	50.	0.26	0.26	0.26
2. Over-pressure impulse	50.	50.	50.	moderate	moderate	moderate
3. Dynamic-pressure peak	50.	50.	50.	moderate	moderate	moderate
4. Dynamic-pressure impulse	50.	50.	50.	moderate	moderate	moderate
5. Under-pressure peak	50.	50.	50.	quite good	quite good	quite good
6. Total thermal energy	50.	50.	50.	quite good	quite good	quite good
7. Maximum irradiance	50.	50.	50.	quite good	quite good	quite good
8. Total dose, tissue	50.	50.	50.	good	good	good
9. Total dose, silicon	51.	52.	55.	quite good	quite good	quite good
10. Total neutron dose	52.	54.	60.	quite good	quite good	quite good
11. Neutron fluence	53.	56.	65.	moderate	moderate	moderate
12. Total gamma dose	54.	58.	70.	moderate	moderate	moderate
13. Minimum threat yield	55.	60.	75.	poor	poor	poor
14. Maximum threat yield	56.	62.	80.	good	poor	very good
15. Ex-atmospheric EMP	57.	64.	85.	moderate	poor	good
16. Endo-atmospheric EMP	58.	66.	90.	poor	poor	good

NAME OF THE SYSTEM: System 3a
Number of elements in the system: 3
Element Nr. 4 is the system.

Parameter	Value (l,c,r)			Nuclear Survivabilities Element 4
1. Over-pressure peak	50.	50.	50.	0.26
2. Over-pressure impulse	50.	50.	50.	moderate
3. Dynamic-pressure peak	50.	50.	50.	moderate
4. Dynamic-pressure impulse	50.	50.	50.	moderate
5. Under-pressure peak	50.	50.	50.	quite good
6. Total thermal energy	50.	50.	50.	quite good
7. Maximum irradiance	50.	50.	50.	quite good
8. Total dose, tissue	50.	50.	50.	good
9. Total dose, silicon	51.	52.	55.	quite good
10. Total neutron dose	52.	54.	60.	quite good
11. Neutron fluence	53.	56.	65.	moderate
12. Total gamma dose	54.	58.	70.	moderate
13. Minimum threat yield	55.	60.	75.	poor
14. Maximum threat yield	56.	62.	80.	good
15. Ex-atmospheric EMP	57.	64.	85.	moderate
16. Endo-atmospheric EMP	58.	66.	90.	poor

MODIFICATIONS

ACTION Nr. 1. Cost increase: medium

Element		Environment parameters Survivability changes [-5,+5]														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1.	3a - 1	2	0	0	0	0	0	0	0	0	0	0	0	2	0	0	1
2.	3a - 2	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1
3.	3a - 3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ACTION Nr. 2. Cost increase: small

Element		Environment parameters Survivability changes [-5,+5]														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1.	3a - 1	2	0	0	0	0	0	0	0	0	0	0	0	2	0	0	2
3.	3a - 3	2	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0

ACTION Nr. 3. Cost increase: large

Element		Environment parameters Survivability changes [-5,+5]														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2.	3a - 2	2	2	2	2	0	0	0	0	0	0	4	4	3	3	3	3
3.	3a - 3	2	2	3	3	0	0	0	0	0	0	4	4	3	0	0	0

No. of Copies	Organization
2	Administrator Defense Technical Info Center ATTN: DTIC-DDA Cameron Station Alexandria, VA 22304-6145
1	Commander U.S. Army Materiel Command ATTN: AMCAM 5001 Eisenhower Ave. Alexandria, VA 22333-0001
1	Director U.S. Army Research Laboratory ATTN: AMSRL-OP-CI-AD, Tech Publishing 2800 Powder Mill Rd. Adelphi, MD 20783-1145
1	Director U.S. Army Research Laboratory ATTN: AMSRL-OP-CI-AD, Records Management 2800 Powder Mill Rd. Adelphi, MD 20783-1145
2	Commander U.S. Army Armament Research, Development, and Engineering Center ATTN: SMCAR-IMI-I Picatinny Arsenal, NJ 07806-5000
2	Commander U.S. Army Armament Research, Development, and Engineering Center ATTN: SMCAR-TDC Picatinny Arsenal, NJ 07806-5000
1	Director Benet Weapons Laboratory U.S. Army Armament Research, Development, and Engineering Center ATTN: SMCAR-CCB-TL Watervliet, NY 12189-4050
1	Director U.S. Army Advanced Systems Research and Analysis Office (ATCOM) ATTN: AMSAT-R-NR, M/S 219-1 Ames Research Center Moffett Field, CA 94035-1000

No. of Copies	Organization
1	Commander U.S. Army Missile Command ATTN: AMSMI-RD-CS-R (DOC) Redstone Arsenal, AL 35898-5010
1	Commander U.S. Army Tank-Automotive Command ATTN: AMSTA-JSK (Armor Eng. Br.) Warren, MI 48397-5000
1	Director U.S. Army TRADOC Analysis Command ATTN: ATRC-WSR White Sands Missile Range, NM 88002-5502
(Class. only) 1	Commandant U.S. Army Infantry School ATTN: ATSH-CD (Security Mgr.) Fort Benning, GA 31905-5660
(Unclass. only) 1	Commandant U.S. Army Infantry School ATTN: ATSH-WCB-O Fort Benning, GA 31905-5000
1	WL/MNOI Eglin AFB, FL 32542-5000
	Aberdeen Proving Ground
2	Dir, USAMSAA ATTN: AMXSY-D AMXSY-MP, H. Cohen
1	Cdr, USATECOM ATTN: AMSTE-TC
1	Dir, ERDEC ATTN: SCBRD-RT
1	Cdr, CBDA ATTN: AMSCB-CII
1	Dir, USARL ATTN: AMSRL-SL-I
5	Dir, USARL ATTN: AMSRL-OP-CI-B (Tech Lib)

No. of Copies	Organization
1	Commander U.S. Army Armament Research, Development, and Engineering Center ATTN: SMCAR-TSS Picatinny Arsenal, NJ 07806-5000
2	Commander U.S. Army Armament Research, Development, and Engineering Center ATTN: SMCAR-FSA SMCAR-FSS Picatinny Arsenal, NJ 07806-5000
1	Commander U.S. Army Armament Research, Development, and Engineering Center ATTN: SMCAR-AET Picatinny Arsenal, NJ 07806-5000
1	Commander U.S. Army Missile Command ATTN: AMSTA-CG Redstone Arsenal, AL 35898-5000
1	CIA 101R/DB/Standard GE47 HQ Washington, DC 20505
1	Director U.S. Army Ballistic Missile Defense Systems Command Advanced Technology Center P. O. Box 1500 Huntsville, AL 35807-3801
1	Commander U.S. Army Watervliet Arsenal ATTN: SARWV-RD, R. Thierry Watervliet, NY 12189-5001
3	Commander U.S. Army Natick R&D Center ATTN: STRNC-OI (3 cps) Natick, MA 01760

No. of Copies	Organization
1	Commander U.S. Army Aviation Systems Command ATTN: AMSAV-ES 4300 Goodfellow Blvd. St. Louis, MO 63120-1798
1	Commander CECOM R&D Technical Library ATTN: ASQNC-ELC-IS-L-R, Meyer Center Fort Monmouth, NJ 07703-5000
1	Director U.S. Army Research Laboratory ATTN: AMSRL-SL White Sands Missile Range, NM 88002-5513
1	Director U.S. Army Research Laboratory ATTN: AMSRL-OP-TL Watertown, MA 02172-0001
1	Director U.S. Army Research Laboratory ATTN: AMSRL-DD-SE 2800 Powder Mill Road Adelphi, MD 20783-1145
1	Commandant U.S. Army Aviation School ATTN: Aviation Agency Fort Rucker, AL 36360
2	Commander U.S. Army Tank-Automotive Command ATTN: SFAE-ASM-SS-T, T. Dean (2 cps) Warren, MI 48397-5000
1	Commander U.S. Army Tank-Automotive Command ATTN: SFAE-ASM-BV Warren, MI 48090-5000
1	Project Manager Abrams Tank System ATTN: SFAE-ASM-AB Warren, MI 48397-5000

No. of Copies	Organization
1	Commander U.S. Army Training & Doctrine Command ATTN: ATCD-MA, MAJ Williams Fort Monroe, VA 23651
1	Commander U.S. Army Research Office ATTN: Technical Library P.O. Box 12211 Research Triangle Park, NC 27709-2211
1	Commander U.S. Army Belvoir R&D Center ATTN: STRBE-WC Fort Belvoir, VA 22060-5606
1	Commander U.S. Army Logistics Management Center Defense Logistics Studies Fort Lee, VA 23801
1	Commandant U.S. Army Command and General Staff College Fort Leavenworth, KS 66027
1	Commandant U.S. Army Special Warfare School ATTN: Rev & Tng Lit Div Fort Bragg, NC 28307
1	Commander U.S. Army Foreign Science & Technology Center ATTN: AMXST-MC-3 220 Seventh Street, NE Charlottesville, VA 22901-5396
2	Commandant U.S. Army Field Artillery Center & School ATTN: ATSF-CO-MW, B. Willis (2 cps) Fort Sill, OK 73503-5600
1	Office of Naval Research ATTN: Code 473, R. S. Miller 800 N. Quincy Street Arlington, VA 22217-9999

No. of Copies	Organization
3	Commandant U.S. Army Armor School ATTN: ATZK-CD-MS, M. Falkovitch (3 cps) Armor Agency Fort Knox, KY 40121-5215
2	Commander Naval Sea Systems Command ATTN: SEA 62R SEA 64 Washington, DC 20362-5101
1	Commander Naval Air Systems Command ATTN: AIR-954, Technical Library Washington, DC 20360
1	Naval Research Laboratory ATTN: Technical Library Washington, DC 20375
1	Commander Dahlgren Division Naval Surface Warfare Center ATTN: Code E23, Technical Library Dahlgren, VA 22448-5000
1	Commander Naval Weapons Center ATTN: Information Science Division China Lake, CA 93555-6001
1	Department of the Navy Naval Ordnance Station Indian Head Detachment ATTN: Technical Library Indian Head, MD 20640-5000
1	OLAC PL/TSTL ATTN: D. Shiplett Edwards AFB, CA 93523-5000
1	WL/MNME Energetic Materials Branch 2306 Perimeter Rd, Ste 9 Eglin AFB, FL 32542-5910

No. of Copies	Organization
1	FTD/NIIS Wright-Patterson AFB, OH 45433
1	Director Lawrence Livermore National Laboratory ATTN: Technical Information Department L-3 P.O. Box 808 Livermore, CA 94550
2	Director Los Alamos National Laboratory ATTN: Document Control for Reports, Library (2 cps) P.O. Box 1663 Los Alamos, NM 87544
1	Director Sandia National Laboratories ATTN: Document Control for 3141, Sandia Report Collection Albuquerque, NM 87115
1	Director Sandia National Laboratories Livermore Laboratory ATTN: Document Control for Technical Library P.O. Box 969 Livermore, CA 94550
1	Director NASA/Scientific & Technical Information Facility P.O. Box 8757 Baltimore/Washington International Airport, MD 21240
1	Aerospace Corporation ATTN: Technical Information Services P.O. Box 92957 Los Angeles, CA 90009
1	The Boeing Company ATTN: Aerospace Library P.O. Box 3707 Seattle, WA 98124-2207

No. of Copies	Organization
1	Director Institute of Defense Analyses ATTN: Library 1801 Beauregard St. Alexandria, VA 22311
1	Battelle ATTN: TACTEC Library, J. N. Huggins 505 King Ave. Columbus, OH 43201-2693
1	Battelle Ordnance Systems and Technology Department ATTN: R. E. Robinson 505 King Ave. Columbus, OH 43201-2693
1	SRI International Propulsion Sciences Division ATTN: Technical Library 333 Ravenswood Ave. Menlo Park, CA 94025-3493
1	Eichelberger Consulting Company ATTN: Dr. R. Eichelberger, President 409 West Catherine Street Bel Air, MD 21014
1	The Johns Hopkins University Applied Physics Laboratory ATTN: Jonathan Fluss Johns Hopkins Road Laurel, MD 20707-0690
1	Pennsylvania State University Department of Mechanical Engineering ATTN: K. Kuo University Park, PA 16802-7501
1	Rensselaer Polytechnic Institute Department of mathematics Troy, NJ 12181
1	Moravian College Department of Mathematics ATTN: M. N. McAllister Bethlehem, PA 18018-6650

No. of Copies	Organization
1	University of Delaware Civil Engineering Department ATTN: S. Kikuchi 137 Dupont Hall Newark, DE 19716
1	Mr. A. Stein 30 Chapel Woods Ct. Williamsville, NY 14221-1816
1	Mr. R. E. Sheer 1916 Bayberry Road Edgewood, MD 21040

No. of Copies	Organization
	Aberdeen Proving Ground
15	Dir, USAMSAA ATTN: AMXSY-A, W. Clifford J. Meredith AMXSY-C, A. Reid AMXSY-CS, P. Beavers C. Cairns D. Frederick AMXSY-G, J. Kramer AMXSY-GA, W. Brooks AMXSY-J, A. LaGrange AMXSY-L, J. McCarthy AMXSY-RA, R. Scungio M. Smith AMXSY-RV, E. Hilkemeyer AMXSY-GS, B. King AMXSY-CC, R. Sandmeyer
2	Cdr, USATECOM ATTN: AMSTE-LFT, D. Gross R. Harrington
1	Cdr, USACSTA ATTN: STECS
2	Cdr, USARDEC ATTN: SMCAR-FSF-T, F. Matts M. Andriolo

INTENTIONALLY LEFT BLANK.

USER EVALUATION SHEET/CHANGE OF ADDRESS

This Laboratory undertakes a continuing effort to improve the quality of the reports it publishes. Your comments/answers to the items/questions below will aid us in our efforts.

1. ARL Report Number ARL-TR-300 Date of Report November 1993

2. Date Report Received _____

3. Does this report satisfy a need? (Comment on purpose, related project, or other area of interest for which the report will be used.) _____

4. Specifically, how is the report being used? (Information source, design data, procedure, source of ideas, etc.) _____

5. Has the information in this report led to any quantitative savings as far as man-hours or dollars saved, operating costs avoided, or efficiencies achieved, etc? If so, please elaborate. _____

6. General Comments. What do you think should be changed to improve future reports? (Indicate changes to organization, technical content, format, etc.) _____

CURRENT ADDRESS

_____
Organization

_____
Name

_____
Street or P.O. Box No.

_____
City, State, Zip Code

7. If indicating a Change of Address or Address Correction, please provide the Current or Correct address above and the Old or Incorrect address below.

OLD ADDRESS

_____
Organization

_____
Name

_____
Street or P.O. Box No.

_____
City, State, Zip Code

(Remove this sheet, fold as indicated, tape closed, and mail.)

(DO NOT STAPLE)